HITACHI
Inspire the Next

TPBroker Version 5

トランザクショナル分散オブジェクト基盤

TPBroker　ユーザーズガイド

解説・手引・文法・操作書

3020-3-U19

マニュアルの購入方法

このマニュアル，および関連するマニュアルをご購入の際は，巻末の「ソフトウェアマニュアルのサービス ご案内」をご参照ください。

■対象製品

●**適用** OS：Windows Server 2003，Windows Server 2008，Windows Vista，Windows XP
P-2464-AF64 Cosminexus TPBroker 05-20
●**適用** OS：AIX 5L V5.3，AIX V6.1
P-1M64-CF61 Cosminexus TPBroker 05-19
●**適用** OS：HP-UX 11i V2，HP-UX 11i V3（IPF）
P-1J64-AR61 Cosminexus TPBroker 05-19
●**適用** OS：Red Hat Enterprise Linux AS 4，Red Hat Enterprise Linux ES 4，Red Hat Enterprise Linux 5
P-9S64-AF61 Cosminexus TPBroker 05-19
●**適用** OS：Red Hat Enterprise Linux AS 4，Red Hat Enterprise Linux 5（IPF）
P-9V64-AF61 Cosminexus TPBroker 05-19
●**適用** OS：Solaris 9，Solaris 10
P-9D64-AF61 Cosminexus TPBroker 05-19
●**適用** OS：Solaris 10
P-9E64-AF61 Cosminexus TPBroker 05-20※
※印が付いているプログラムプロダクトについては，発行時期をご確認ください。
これらのプログラムプロダクトのほかにも，このマニュアルをご利用になれる場合があります。詳細は「リリースノート」でご確認ください。

■輸出時の注意

本製品を輸出される場合には，外国為替および外国貿易法ならびに米国の輸出管理関連法規などの規制をご確認の上，必要な手続きをお取りください。
なお，ご不明な場合は，弊社担当営業にお問い合わせください。

■商標類

■発行

2008 年 11 月 (第 1 版) 3020-3-U19

■著作権

はじめに

このマニュアルは，トランザクショナル分散オブジェクト基盤 TPBroker の概要，機能，運用方法について説明したものです。

■対象読者

システム管理者，システム設計者およびオペレータで，TPBroker を使用して分散オブジェクトコンピューティング環境を構築または運用する方を対象としています。次の知識がある方を対象としています。

- C++ または Java
- OTS
- CORBA

■マニュアルの構成

このマニュアルは，次に示す編から構成されています。

第 1 編　概説

OTS（オブジェクトトランザクションサービス）および TPBroker の概要について説明しています。

第 2 編　環境設定

TPBroker を運用する前に行う環境設定やセットアップの手順について説明しています。

第 3 編　TPBroker の機能

TPBroker の機能のうち，OTS 機能および運用支援機能について説明しています。

第 4 編　運用

TPBroker の開始と終了などの運用方法と，運用方法を定める TPBroker の定義および運用コマンドについて説明しています。

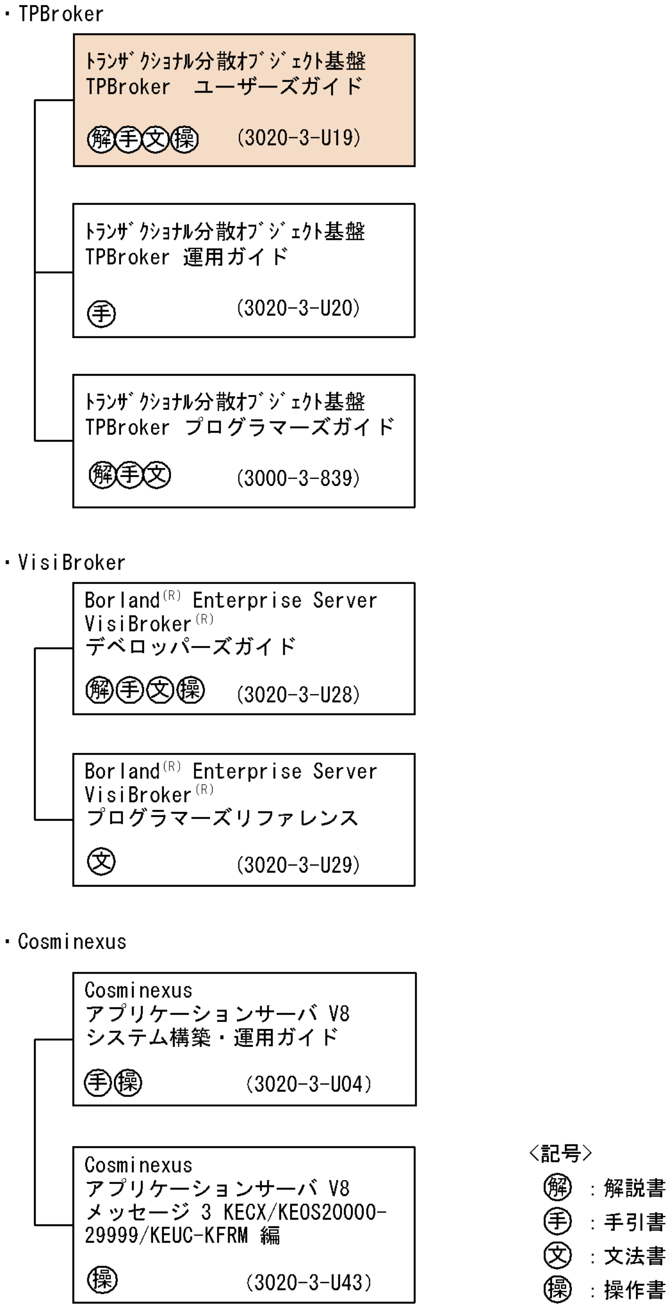
第 5 編　障害対策

TPBroker の障害の内容および障害時に取得される情報について説明しています。

第 6 編　メッセージ

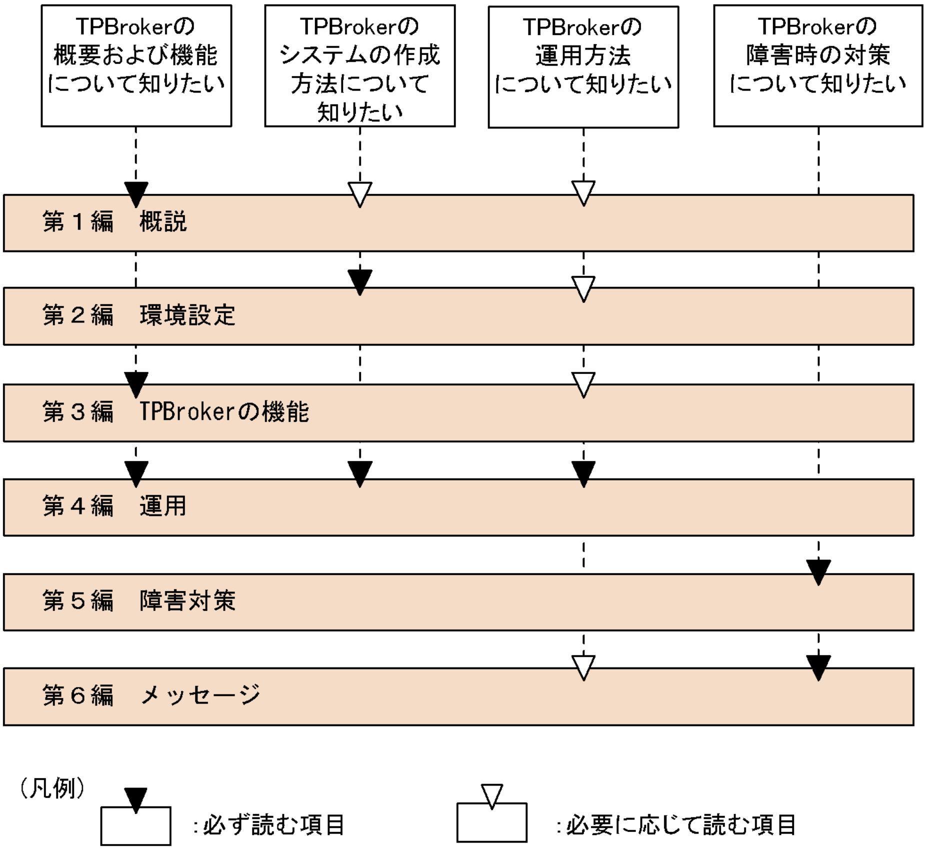
TPBroker および Java OTS が出力するメッセージについて説明しています。

■関連マニュアル



■読書手順

このマニュアルは，利用目的に合わせて直接編を選択して読むことができます。利用目的別に次の流れに従ってお読みいただくことをお勧めします。



■このマニュアルでの表記

このマニュアルでは，製品名称および名称を次に示す略称で表記します。

略称	製品名称および名称
AIX	AIX 5L V5.3
	AIX V6.1
HP-UX（IPF）	HP-UX 11i V2
	HP-UX 11i V3(IPF)
IPF	Itanium(R) Processor Family
Java	Java™
Linux	Red Hat Enterprise Linux(R) AS 4
	Red Hat Enterprise Linux(R) AS 4 (IPF)
	Red Hat Enterprise Linux(R) ES 4
	Red Hat Enterprise Linux(R) ES 4 (IPF)
	Red Hat Enterprise Linux(R) 5

略称	製品名称および名称
	Red Hat Enterprise Linux(R) 5 (IPF)
Solaris	Solaris 9
	Solaris 10
VisiBroker	Borland(R) Enterprise Server VisiBroker(R)
Windows Server 2003	Microsoft(R) Windows Server(R) 2003, Enterprise Edition
	Microsoft(R) Windows Server(R) 2003, Standard Edition
	Microsoft(R) Windows Server(R) 2003 R2, Enterprise Edition
	Microsoft(R) Windows Server(R) 2003 R2, Standard Edition
Windows Server 2008	Microsoft(R) Windows Server(R) 2008, Standard Edition
	Microsoft(R) Windows Server(R) 2008, Enterprise Edition
Windows Vista	Microsoft(R) Windows Vista(R) Business
	Microsoft(R) Windows Vista(R) Enterprise
	Microsoft(R) Windows Vista(R) Ultimate
Windows XP	Microsoft(R) Windows(R) XP Professional Operating System

Windows Server 2003，Windows Server 2008，Windows Vista，および Windows XP で機能差がない場合，Windows と表記しています。
また，下記に示すプログラムプロダクトで仕様差がない場合，TPBroker と表記しています。

- Cosminexus TPBroker
- Cosminexus TPBroker Developer
- TPBroker
- TPBroker Developer

また，ご使用になるプログラミング言語または OS によって説明が異なる場合，次の記号を使用しています。

記号	意味
(C++)	C++ 言語，または TPBroker で提供する C++ インタフェースを使用するアプリケーションプログラムに該当
(Cosminexus TPBroker)	Cosminexus TPBroker に該当
(Java)	Java 言語，または TPBroker で提供する Java インタフェースを使用するアプリケーションプログラムに該当
(UNIX)	すべての UNIX プラットフォームに該当
(Windows)	Windows に該当

なお，環境変数の設定を説明する文中の「$」は UNIX での表記であり，Windows の場合は「% ～ %」，同様にディレクトリの区切り文字の「/」は UNIX での表記であり，Windows の場

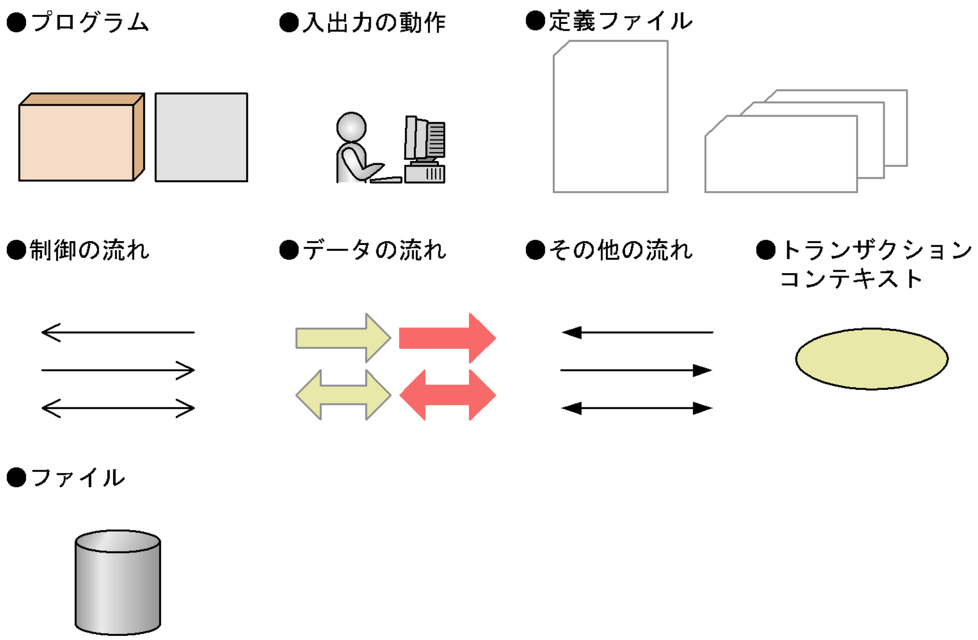
合は「¥」となります。

このマニュアルで使用する英略語の一覧を示します。

英略語	英字での表記
ADM	Administration
API	Application Programming Interface
CORBA	Common Object Request Broker Architecture
DB	Database
DBMS	Database Management System
DLL	Dynamic Linking Library
DTP	Distributed Transaction Processing
EB	Enterprise Bean
EJB	Enterprise JavaBeans™
GUI	Graphical User Interface
IDL	Interface Definition Language
IIOP	Internet Inter-ORB Protocol
IOR	Interoperable Object Reference
J2EE	Java 2 Enterprise Edition
JSP	Java Server Pages
OAD	Object Activation Daemon
OMG	Object Management Group
ORB	Object Request Broker
OS	Operating System
OTS	Object Transaction Service
OTSCRM	Object Transaction Service Communication Resource Manager
POA	Portable Object Adapter
RMI	Java Remote Method Invocation
SQL	Structured Query Language
TCP	Transmission Control Protocol
TCS	Transaction Context Server
UAP	User Application Program
VM	Virtual Machine

■図中で使用する記号

このマニュアルの図中で使用する記号を，次のように定義します。



■文法の記号

このマニュアルで使用する記号の意味を示します。

ただし，C++ 言語および Java 言語のインタフェースやコーディング例の説明は，それぞれの言語の文法規則に従います。これらの記号の意味は適用されません。

<table>
<tr><th>文法記述記号</th><th>意味</th></tr>
<tr><td>[ ]</td><td>この記号で囲まれている項目は省略してもよいことを表します。
（例）admstart [-f]
これは，admstart と指定するか，または admstart -f と指定することを表します。</td></tr>
<tr><td>|</td><td>この記号で区切られた項目を選択できることを表します。
（例）admstop -f 1|2
これは，-f オプションに 1 か 2 のどちらかを指定できることを表します。</td></tr>
<tr><td>{ }</td><td>この記号で囲まれている複数の項目のうちから一つを選択することを表します。
（例）{-t|-T トランザクショングローバル識別子}
これは，-t と -T トランザクショングローバル識別子のうち，どちらかを指定することを表します。</td></tr>
<tr><td>…</td><td>この記号で示す直前の項目を繰り返し指定できることを表します。
（例）tslsrm -o ファイル名 [, ファイル名]...
これは，-o オプションのファイル名を繰り返し指定できることを表します。</td></tr>
<tr><td>～</td><td>この記号のあとにユーザ指定値の属性を表します。</td></tr>
</table>

文法記述記号	意味
…	この記号は文字列を省略していることを表します。 （例）スペースを含むパスを指定する場合は，パス全体を「¥"…¥"」のように「¥"」で囲んでください。 これは，「…」部分で「C:¥Program Files¥Sample¥sample.txt」といったパス名（文字列）を省略していることを表します。
△	半角スペースを表します。
<<　>>	ユーザが指定を省略したときの省略値を表します。
<　>	ユーザ指定値の構文要素を表します。
((　))	ユーザ指定値の指定範囲を表します。
< 英数字 >	英字と数字（0 ～ 9）で指定することを表します。
< 文字列 >	任意の文字の配列で指定することを表します。
< パス名 >	記号名称，¥，/，および .（ピリオド）で指定することを表します。 ただし，パス名は使用する OS に依存します。

■常用漢字以外の漢字の使用について

このマニュアルでは，常用漢字を使用することを基本としていますが，次に示す用語については，常用漢字以外の漢字を使用しています。
個所（かしょ）　同梱（どうこん）　必須（ひっす）

■ KB（キロバイト）などの単位表記について

1KB（キロバイト），1MB（メガバイト），1GB（ギガバイト），1TB（テラバイト）はそれぞれ 1,024 バイト，$1,024^{2}$ バイト，$1,024^{3}$ バイト，$1,024^{4}$ バイトです。

目次

第 1 編　概説

第 2 編　環境設定

第 3 編　TPBroker の機能

第 4 編　運用

第 5 編　障害対策

図目次

表目次

1 TPBroker の概要

TPBroker は，分散オブジェクトコンピューティング環境でのオブジェクト間の通信やトランザクション制御，および運用支援機能を提供する製品です。
この章では，TPBroker の概要について説明します。

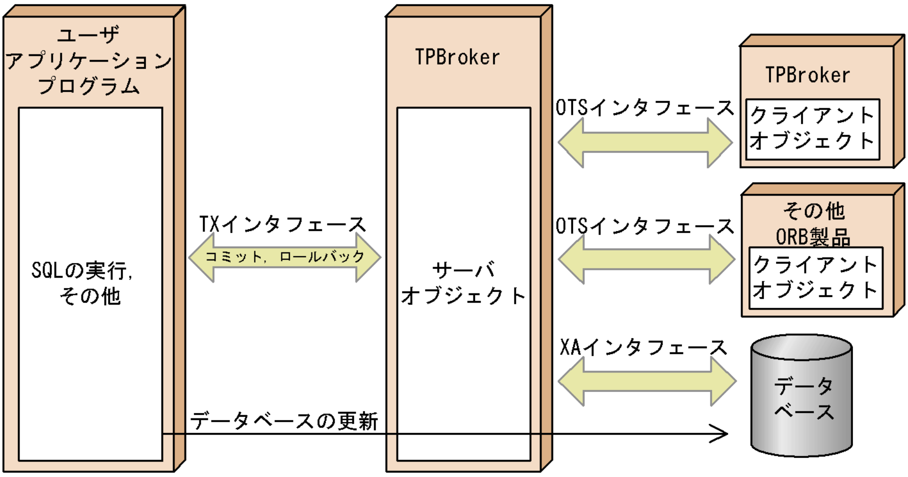
1.1 TPBroker とは

TPBroker は，分散オブジェクトコンピューティング環境でのオブジェクト間の通信やトランザクション制御，および運用支援機能を提供する製品です。

TPBroker を利用すると，既存のシステム資源を新しいシステムに活用できます。つまり，既存のシステムはそのままに，分散オブジェクト環境のインタフェースを追加するだけで分散オブジェクトを適用した新しいシステムを構築できます。既存システムが複雑で変更が難しい場合は特に有効です。

TPBroker が提供する分散オブジェクトコンピューティング環境のインタフェースを，次の図に示します。

図 1-1　TPBroker が提供する分散オブジェクトコンピューティング環境のインタフェース



1.1.1 CORBA と IIOP

TPBroker は，オブジェクト指向技術の普及と標準化のための非営利団体である OMG が提案する CORBA の仕様に基づいています。CORBA で規定する分散オブジェクトコンピューティング環境は国際標準規格に基づいており，今後の重要なコンピュータアーキテクチャ規格になると考えられています。そのため，TPBroker を使用することによって，拡張性と将来性がある新しいシステムを構築できます。

また，TPBroker では，CORBA の ORB 機能を使用して，分散オブジェクト指向のアプリケーションプログラムを開発できます。そのため，システム資源の再利用やプログラム変更の局所化ができ，システム開発や管理コストの低減などが期待できます。

さらに，TPBroker では，通信プロトコルとして CORBA で規定された IIOP をサポートしています。

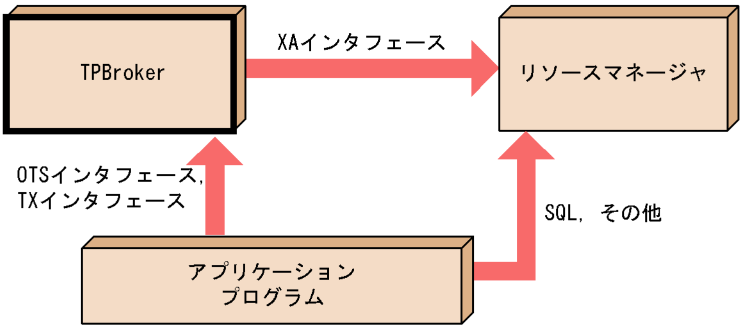
1.1.2　OTS と X/Open 標準インタフェースのサポート

TPBroker では，CORBA で規定された OTS の仕様をサポートしています。TPBroker Version 5 は OTS 1.3 の仕様に準拠しています。OTS のサポートによって，マルチサーバ間での負荷分散や，障害に対応したフォールトトレランスができます。そのため，高信頼および高性能の分散コンピューティングシステムを構築できます。

C++ 言語でアプリケーションプログラムを開発した場合

X/Open で規定された分散トランザクションのための標準仕様である DTP モデルのインタフェースをサポートしています。そのため，CORBA に準拠したほかの ORB 製品，および X/Open の DTP モデルに準拠したほかの製品と相互接続できます。したがって，ユーザは TX インタフェースを使用したトランザクションの制御ができます。また，リソースマネージャが XA インタフェースをサポートしていれば，TPBroker のトランザクションと同期を取ることができます。
TPBroker とアプリケーションプログラムの関係を，次の図に示します。

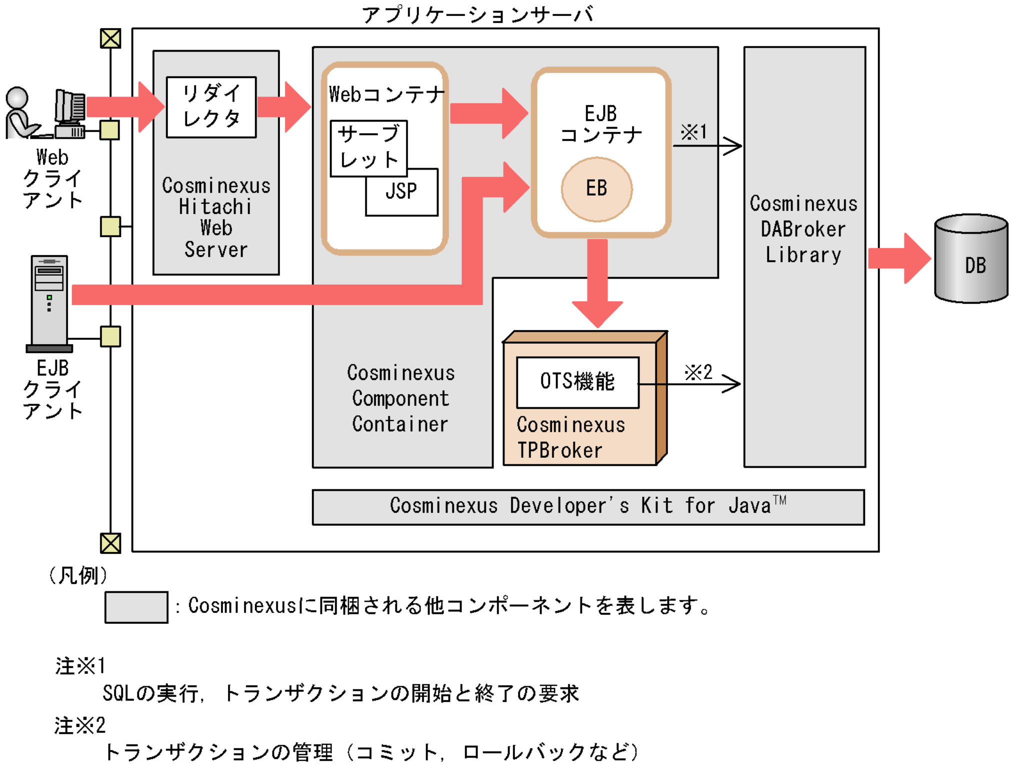
図 1-2　TPBroker とアプリケーションプログラムの関係



1.1.3　Cosminexus TPBroker とは

Cosminexus TPBroker は Cosminexus 用の TPBroker です。Cosminexus TPBroker を使用すると，RMI-IIOP 通信基盤とともに TPBroker のトランザクション機能（OTS）を利用することで，J2EE サーバ上での分散トランザクション処理を実現できます。
Cosminexus TPBroker の位置づけを次の図に示します。

図 1-3　Cosminexus TPBroker の位置づけ



Cosminexus TPBroker をご使用の場合の参照個所

Cosminexus TPBroker をご使用の場合，次の表に従ってマニュアルをお読みください。種別が「○」になっている項目を，適宜参照してください。

表 1-1　Cosminexus TPBroker をご使用の場合の参照個所

記載個所	目次タイトル	種別
1.1	TPBroker とは	○
1.2	TPBroker の特長	○※1
1.3	TPBroker の機能	○
2.1	環境設定の手順	○
2.2	環境変数を設定する	○
2.3	TPBroker の OTS 環境をセットアップする	○
2.4	システム環境定義を変更する	○
2.5	リソースマネージャと連携する場合の準備（C++）	×
2.6	プロセス監視定義ファイルを編集する	○
2.7	TPBroker の運用支援機能実行環境をセットアップする	○

記載個所	目次タイトル	種別
3.1	トランザクション制御	△
3.1.1	CORBA で規定された OTS の仕様	△
3.1.2	トランザクション制御の概要	△
3.1.3	オブジェクトトランザクションサービス	△
3.1.4	トランザクションモデル	△
3.1.5	コンテキスト管理	△
3.1.6	プロパゲーション	△
3.1.7	チェックドトランザクション	△
3.1.8	ポリシーの設定	△
3.1.9	トランザクション稼働統計情報	○
3.1.10	トランザクショントレース	○
3.2	回復処理	○
4	C++ OTS 機能（C++）	×
5	Java OTS 機能（Java）	△
6.1	システム運用	○
6.2	プロセス監視	○
6.3	監視対象プロセス並列起動／停止機能	○
6.4	プロセス監視定義ファイルの再読み込み機能	○
6.5	メッセージログの管理	○
6.6	稼働統計情報の取得	○
6.7	UAP ログ出力機能（C++）（UNIX）	×
7.1	TPBroker の運用の流れ	○
7.2	TPBroker の開始と終了	○
7.3	アプリケーションプログラムの開始と終了	×
7.4	リソースマネージャの運用（C++）	×
7.5	トランザクションサービスの運用	○
7.6	TPBroker ファイルシステム（UNIX）	○
7.7	トランザクショントレースの運用（UNIX）	○
7.8	TPBroker デーモン	○
7.9	TPBroker のバージョンアップ	×
8.1	定義の概要	○
8.2	プロセス監視定義の詳細	○
8.3	システム環境定義の詳細	○
8.3.1	運用定義	○

記載個所	目次タイトル	種別
8.3.2	トランザクション定義	○※2
8.3.3	リソースマネージャ定義（C++）	×
8.3.4	回復定義	○
8.3.5	トランザクションコンテキストサーバ定義	×
8.3.6	システム定義	○
8.3.7	トランザクショントレース定義	○
8.4.1	リソースマネージャとの XA 連携（C++）	×
8.4.2	リソースマネージャの削除（C++）	×
8.4.3	プロセス監視定義の定義例	○
9.1	運用コマンドの概要	○
9.2	TPBroker で使用する運用コマンド	○※3
9.3	運用コマンドの詳細	○※3
10.1	アプリケーションプログラムの障害	△
10.2	TPBroker の障害	○
10.3	障害の解決に必要な情報	○
11.	メッセージ	○
12.	Java OTS が出力するメッセージ	○

（凡例）

○

Cosminexus TPBroker でサポートしている機能についての記載個所です。Cosminexus TPBroker をご使用の場合，参照してください。

×

Cosminexus TPBroker ではサポートしていない機能についての記載個所です。Cosminexus TPBroker をご使用の場合，参照する必要はありません。

△

Cosminexus TPBroker，またはほかの Cosminexus 製品でサポートしているが，ユーザが意識する必要はない機能についての記載個所です。Cosminexus TPBroker をご使用の場合，参照する必要はありません。

注※1

CORBA アプリケーションの開発／実行は，Cosminexus TPBroker ではサポートしていません。

注※2

コンパイラ選択は，ユーザが意識する必要はない機能についての記載個所です。

注※ 3

次に示すコマンドは，Cosminexus TPBroker ではサポートしていません。

- tsedapt（API トレースファイル解析（C++））
- tslnkrm（リソースマネージャの登録・削除（C++））
- tslsrm（リソースマネージャ情報の表示（C++））
- tsmkobj（トランザクション制御用オブジェクトファイルの作成（C++））

1.2 TPBroker の特長

TPBroker には，次のような特長があります。

● マルチプラットフォームに対応
TPBroker は，マルチプラットフォームに対応しています。TPBroker で対応しているプラットフォームを，次に示します。

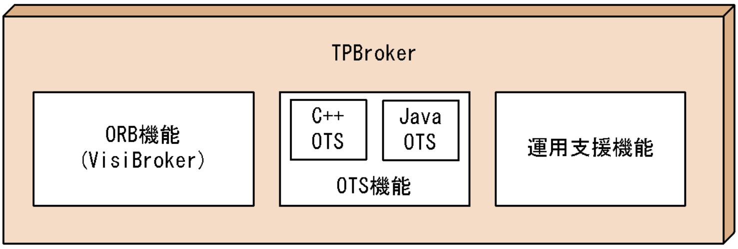
- Windows
- HP-UX
- AIX
- Solaris
- Linux

● C++ 言語，Java 言語に対応
TPBroker が提供するインタフェースは，CORBA の C++ 言語および Java 言語のマッピングをサポートしています。そのため，C++ 言語および Java 言語によるオブジェクト開発ができます。
CORBA アプリケーションの開発／実行は，Cosminexus TPBroker ではサポートしていません。

1.3 TPBroker の機能

TPBroker は，ネットワーク上でオブジェクトを管理したりオブジェクト間の通信をしたりする ORB 機能，トランザクションを制御する OTS 機能，および運用支援機能の三つの機能から構成されています。TPBroker の機能の構成について次の図に示します。

図 1-4　TPBroker の機能の構成



TPBroker はこれらの三つの機能を組み合わせて運用します。OTS 機能，または運用支援機能を使用しないで運用することもできます。

それぞれの機能の概要を次に説明します。

1.3.1 ORB 機能

TPBroker の ORB 機能では，オブジェクトの管理，オブジェクト間の通信などネットワーク上のオブジェクトの相互作用を管理します。TPBroker では，ORB 機能として CORBA 2.5 に準拠した製品である VisiBroker を使用しています。そのため，VisiBroker で実現できる分散オブジェクトの管理，および分散オブジェクトの通信ができます。ここでは，TPBroker が提供する ORB 機能の詳細について説明します。

VisiBroker の主な機能については，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

(1) スマートエージェントによる分散オブジェクトの管理

VisiBroker のスマートエージェント機能（OSAgent）では，サーバ上で複数のエージェントを稼働してオブジェクトを管理できます。エージェントに障害が発生した場合，そのエージェントが管理していたオブジェクトは自動的にほかのオブジェクトに再登録されるため，オペレーションは中断しません。

また，エージェントがネットワーク上のオブジェクトを動的に管理するため，クライアントごとにオブジェクトの実装を意識する必要はありません。オブジェクトの事前定義も不要です。

(2) スマートバインディングによる分散オブジェクト間の通信

VisiBroker のスマートバインディング機能では，クライアントからサーバ上のオブジェクトにアクセスすると，そのオブジェクトがクライアントと同一プロセスにあるか，同一ノードの別プロセスにあるか，またはリモートノードにあるかを判断し，自動的に最適な通信機能を選択します。このため，通信時のオーバヘッドが削減でき，高速な処理を実現できます。

例えば，クライアントとサーバ上のオブジェクトが同一のプロセスで実行される場合は，ORB を介さないで実行します。また，同一ノードの別プロセスにある場合は，クライアントとサーバ間の情報の受け渡しに共有メモリを利用します。

(3) マルチスレッドによる処理

VisiBroker のマルチスレッド機能では，一つのプロセスで複数のオブジェクトからの要求を同時に処理できます。そのため，サーバのリソースを有効に利用でき，システムの性能とスケーラビリティを向上できます。

(4) IIOP による相互接続

VisiBroker では，通信プロトコルとして CORBA で規定された IIOP をサポートしています。そのため，TPBroker と VisiBroker との接続，および CORBA に準拠しているほかの ORB 製品との接続ができます。また，インターネット，イントラネット，およびエクストラネットを前提とした新しいシステムを構築できます。

(5) CORBA の仕様

(a) オブジェクトモデル

CORBA のオブジェクトは IDL で定義されます。IDL によってその継承関係，クラスが提供するメソッドのパラメタ，リターン値とメソッド内で発生する例外を定義できます。

クライアントは，CORBA オブジェクトの代理オブジェクトであるオブジェクトリファレンスにリクエストを発行し，CORBA オブジェクトをアクセスできます。オブジェクトリファレンスの形式は ORB によって異なりますが，TPBroker では C++ および Java オブジェクトのポインタとして定義されています。

(b) オブジェクトインプリメンテーション

クライアントからのリクエストを処理するサーバをオブジェクトインプリメンテーションと呼びます。オブジェクトインプリメンテーションは，IDL で定義された CORBA オブジェクトの実体であり，TPBroker では C++ および Java オブジェクトを含むプロセスとして定義されています。

(c) オブジェクトリクエストブローカ（ORB）

ORB はクライアント／サーバ間の通信，およびオブジェクトインプリメンテーションの起動を提供します。オブジェクトのライフサイクルのサービスは，上位サービスとして

規定され，ORB には含まれません。

1.3.2 OTS 機能

TPBroker が提供するトランザクション制御機能（OTS 機能）は，CORBA で規定された OTS 1.3 の仕様に準拠しています。TPBroker のユーザインタフェースを使用すると，2 相コミットによってトランザクション処理を制御できます。

OTS の機能の詳細は，「3. OTS 機能」を参照してください。

(1) トランザクション制御

2 相コミットとは，同期点での処理をプリペア処理（資源のアップデート準備）とコミット処理（資源のアップデート処理）という 2 段階に分ける方法です。

2 相コミットでは，DBMS などの複数のリソースオブジェクトに対して同期を取り，コミットまたはロールバックができます。また，2 相コミットではトランザクション処理中に障害が発生した場合でも，すべてのリソースオブジェクトを矛盾なく自動的にロールバックできます。

(2) マルチスレッドアプリケーションのサポート

TPBroker は C++ および Java のマルチスレッド環境に対応した，完全にスレッドセーフなトランザクション機能をサポートしています。

(3) 回復処理

TPBroker では，システムの一部に障害が発生しても，部分回復処理をしてシステムが全面停止しないようにしています。また，システムが全面停止しても全面回復処理をして，システムの状態を回復させます。

1.3.3 C++ OTS の機能

C++ OTS は，OTS のインタフェースだけでなく，X/Open で規定された分散トランザクションのための標準仕様である DTP モデルの TX インタフェース，および XA インタフェースをサポートします。そのため，XA インタフェースを備えた Oracle などの DBMS をはじめとしたリソースマネージャと連携してトランザクションを制御できます。

(1) リソースマネージャとの連携

C++ アプリケーションは，リソースマネージャと連携してトランザクションを処理できます。TPBroker は OTS のインタフェースだけでなく，X/Open の DTP モデルの TX インタフェースと XA インタフェースをサポートしているため，OTS の仕様をサポートしていないリソースマネージャでも，XA インタフェースをサポートしていれば C++ アプリケーションのトランザクション処理と同期を取ることができます。

ただし，使用するリソースマネージャが XA インタフェースをサポートしている場合でも，XA インタフェースを使って C++ アプリケーションと連携していない場合は同期を取れません。また，リソースマネージャが XA インタフェースをサポートしていない場合，リソースマネージャへアクセスはできますが，C++ アプリケーションのトランザクション処理と同期は取れません。

Java アプリケーションでリソースマネージャと連携するには，Java アプリケーションとリソースマネージャを中継する C++ アプリケーションが必要です。

(2) 時間監視機能

TPBroker は，トランザクションの処理時間を監視し，指定した時間を過ぎた場合の対処を提供しています。

1.3.4 Java OTS の機能

Java でアプリケーションプログラムを開発すると，Java OTS の機能を使用できます。

(1) OMG OTS 仕様の Java サポート

Java OTS は，C++ に関連するサポート（C++ 実行環境が提供するサポート）以外の OTS 1.3 インタフェースを完全にサポートします。これには，次のものが含まれます。

- すべての OTS オブジェクトおよび例外についてのサポート
- トランザクショナル Java アプリケーションを作成するための OTS クラスライブラリ
- ネスティッドトランザクション

Java OTS は，直接コンテキスト管理および間接コンテキスト管理の両方をサポートします。OTS のインタフェース定義を使って直接コンテキスト管理および間接コンテキスト管理の両方をサポートする OTS を呼び出します。

Java OTS は，暗黙的プロパゲーションおよび明示的プロパゲーションの両方をサポートします。

なお，次に示す OTS インタフェースは Java 言語をサポートしていません。

- X/Open TX インタフェース
- リソースマネージャへの接続に関する X/Open XA 仕様

Java アプリケーションで TX または XA インタフェースにアクセスするには，Java アプリケーションとこれらのインタフェース間を中継する C++ アプリケーションを作成する必要があります。

(2) ライトウェイトライブラリ

Java OTS の中心となる機能は，ライトウェイトクライアント／サーバライブラリです。Java アプリケーションを実行するマシンには，デーモンや固定的な記憶領域は必要あり

ません。OTS 環境の一部として，サーバマシン上のトランザクションコンテキストサーバデーモンが，トランザクショナル Java アプリケーションから任されたすべてのトランザクションコンテキストを管理します。

(3) C++ で開発したアプリケーションプログラムとの相互運用性

Java で開発したアプリケーションプログラムには，C++ で開発したアプリケーションプログラムとの完全な相互運用性があります。Java アプリケーションは，TPBroker OTS を使用する C++ アプリケーションにトランザクションをプロパゲーションすることができます。また，C++ アプリケーションは Java アプリケーションにトランザクションをプロパゲーションすることができます。

一般的に，Java で開発したクライアントアプリケーションは C++ で開発したサーバアプリケーションをトランザクションとともに呼び出します。C++ で開発したサーバアプリケーションは，X/Open XA インタフェースを使用してデータベースにアクセスします。そして，クライアントアプリケーションはトランザクションをコミットします。

Java OTS は，同一プロセス上での，C++ で開発したアプリケーションプログラムと Java で開発したアプリケーションプログラムの混在をサポートしていません。例えば，Java アプリケーション上で Java Native Interface（JNI）を使用して C++ ライブラリを呼び出し，C++ の API を使用することはできません。Java と C++ が混在するプログラムを使用しても，Java OTS は Java プログラム上で C++ の API をサポートしません。

Java のコーディングは ORB の通信を介して C++ で開発したプログラムと協調します。このため，各言語のプログラムをシンプルにできます。

1.3.5 運用支援機能

TPBroker の運用支援機能では，システム運用，プロセス監視，メッセージログ管理，および稼働統計情報の取得などの機能を提供しています。そのため，システム運用時のオペレータの操作は省力化され，システムの運用コストを削減できます。

運用支援機能の詳細については，「6. 運用支援機能」を参照してください。

(1) システム運用

TPBroker の開始および終了，TPBroker を運用するために必要な各種運用コマンド，および稼働統計情報の表示など，システム運用全般に関する支援を提供しています。

(2) プロセス監視

TPBroker では，幾つかのシステムデーモンプロセスを提供しています。TPBroker はこれらのプロセスの状態やリソースマネージャを監視し，障害の発生を自動的に検知します。また，TPBroker では，システム環境に応じてプロセス監視方式を選択できます。例えば，あらかじめプロセス監視定義ファイルに監視対象となるプロセスを設定しておく方式や，TPBroker の運用コマンドで起動したプロセスを動的に監視対象に加える方式な

どがあります。

(3) メッセージログ管理

TPBroker では，メッセージをイベントログ，メッセージログファイル，および syslog に出力します。

(4) 稼働統計情報の取得

TPBroker では，稼働統計情報を取得します。取得した情報は運用コマンドを使用して表示できます。

2 TPBroker の環境設定

この章では，TPBroker の環境設定を手順に従って説明します。

2.1 環境設定の手順

TPBroker の環境設定の手順を次に示します。高速オプションライブラリ（OTS Fast Path Option）を使用する場合も同じ手順です。

1. **TPBroker をインストールする（必須）**
   インストールの手順の詳細については，「ソフトウェア添付資料」または「TPBroker 使用時の注意事項　取扱説明書」を参照してください。
   また，UNIX 版の場合 syslog 出力するための設定も実施してください。
   TPBroker は syslog にメッセージを書き込みます。user 機能の info レベル以上のログを保存するように設定してください。

   例　/etc/syslog.conf に以下の行を追加します。

   ```
   user.info             /var/log/syslog
   ```

2. **環境変数を設定する（必須）**
   TPBroker を使用するために必要な環境変数を設定します。ORB の環境変数や localaddr などの設定については，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

3. **TPBroker の OTS 環境をセットアップする（必須）**
   tssetup コマンド（Windows 版 05-19 以降では tstpbsetup コマンド。以下本マニュアルでは tssetup に表記を統一します）で TPBroker の OTS 環境をセットアップします。

4. **システム環境定義を変更する（任意）**
   セットアップによって設定されたシステム環境定義の値を，必要に応じて変更します。

5. **リソースマネージャと連携する場合の準備をする（C++）（任意）**
   XA インタフェースをサポートしたリソースマネージャを TPBroker と連携して使用する場合の準備をします。この手順は C++ 実行環境の場合だけ必要です。

6. **プロセス監視定義ファイルを編集する（任意）**
   プロセス監視を実行するための定義ファイルに，必要な項目を設定します。

7. **TPBroker の運用支援機能実行環境をセットアップする（任意）**
   admsetup コマンドで TPBroker の運用支援機能実行環境をセットアップします。

2.2 環境変数を設定する

TPBroker を使用するために，次に示す環境変数を設定してください。システム環境変数の値を変更した場合は，Windows を再起動してください。

なお，説明中の「$」で始まる文字列は UNIX での表記であり，Windows の場合は「%」で始まり「%」で終わる文字列になります。同様に「/」は UNIX の表記であり，Windows の場合は「¥」になります。例えば，UNIX で $TPDIR/lib の場合，Windows では %TPDIR%¥lib になります。

(1) 必ず設定する環境変数

● CLASSPATH（Java）

JAR ファイルを環境変数の値に追加します。

Java OTS 用 JAR ファイル

```
$TPDIR/lib/tpcosots.jar
```

ORB 用 JAR ファイル

```
$TPDIR/lib/vbjorb.jar
```

（Windows）

```
prompt> set
CLASSPATH=%TPDIR%¥lib¥tpcosots.jar;%TPDIR%¥lib¥vbjorb.jar
```

（UNIX）

csh（C シェル）の場合

```
prompt> setenv CLASSPATH $TPDIR/lib/tpcosots.jar:$TPDIR/lib/
vbjorb.jar
```

sh（ボーンシェル）の場合

```
prompt> CLASSPATH=$TPDIR/lib/tpcosots.jar:$TPDIR/lib/
vbjorb.jar
prompt> export CLASSPATH
```

● OSAGENT_PORT

OSAgent が使用するポート番号を指定します。デフォルト値は 14000 です。
OSAgent は起動時に，定義されたポート番号でリクエストを検出しようとします。TPBroker は，この環境変数からポート番号を取得して，そのポート番号によって OSAgent と通信します。
一つのホストで複数の OSAgent プロセスを実行する場合，各 OSAgent プロセスには，それぞれ別のポート番号を指定してください。
ポート番号を 10000 にする例を次に示します。

（Windows）

```
prompt> set OSAGENT_PORT=10000
```

（UNIX）

csh（C シェル）の場合

```
prompt> setenv OSAGENT_PORT 10000
```

sh（ボーンシェル）の場合

```
prompt> OSAGENT_PORT=10000
prompt> export OSAGENT_PORT
```

● PATH

TPBroker が提供する運用コマンドの格納ディレクトリ（$TPDIR/bin）を指定します。

（Windows）

```
prompt> set PATH=%TPDIR%¥bin;%PATH%
```

（UNIX）

csh（C シェル）の場合

```
prompt> setenv PATH $TPDIR/bin:$PATH
```

sh（ボーンシェル）の場合

```
prompt> PATH=$TPDIR/bin:$PATH
prompt> export PATH
```

● TPDIR

TPBroker がインストールされたディレクトリを指定します。

● TPJDIR（Java）

TPBroker がインストールされたディレクトリを指定します。

● TZ

タイムゾーンを指定します。日本時間の場合は，JST-9 を指定してください。

● VBROKER_ADM

TPBroker のインタフェースリポジトリ，OAD，および OSAgent の定義情報が格納されているディレクトリを指定します。
環境変数 VBROKER_ADM には，$TPDIR/adm を設定することをお勧めします。
$TPDIR/adm 以外を設定する場合は，$TPDIR/adm をディレクトリごとコピーした上で，この環境変数を設定してください。
またこの環境変数は，デフォルトでは ORB のトレースファイルなどの出力先の起点になります。ORB のトレースファイルなどの出力先は，デフォルトでは $VBROKER_ADM/../log（C++）および $VBROKER_ADM/../logj（Java）になります。

（Windows）

```
prompt> set VBROKER_ADM=%TPDIR%¥adm
```

（UNIX）

csh（C シェル）の場合

```
prompt> setenv VBROKER_ADM $TPDIR/adm
```

sh（ボーンシェル）の場合

```
prompt> VBROKER_ADM=$TPDIR/adm
prompt> export VBROKER_ADM
```

次に示す三つの環境変数は，使用する OS によって異なります。共用ライブラリのパスに，TPBroker が提供するライブラリのディレクトリ（$TPDIR/lib）を追加してください。これらの環境変数は OS ごとに必ずアップデートしてください。C++ 実行環境の場合，tslnkrm コマンドに -i オプションを付けて，リソースマネージャを TPBroker に登録しているときは，$TPSPOOL/XA も追加してください。

- SHLIB_PATH（HP-UX 11.0，11i，11i V2（PA-RISC）の場合）
- LD_LIBRARY_PATH（Solaris，Linux，HP-UX（IPF）の場合）
- LIBPATH（AIX の場合）

(2) オプションの環境変数

- ADMFS

  TPBroker の運用支援機能の情報（システムステータスファイル）を格納するディレクトリ，またはキャラクタ型スペシャルファイルを指定します。
  admsetup コマンドに -c オプションを付けて実行したときに，この環境変数に設定されているディレクトリが生成されます。
  キャラクタ型スペシャルファイルを指定する場合は，あらかじめ tsmkfs コマンドで TPBroker ファイルシステムを作成しておく必要があります。TPBroker ファイルシステムについては「7.6　TPBroker ファイルシステム（UNIX）」を参照してください。
  この環境変数が設定されていない場合は，環境変数 ADMSPOOL に指定したディレクトリが使用されます。
  同一ノード内に複数の TPBroker の運用支援機能の実行環境を登録する場合は，それぞれの環境で異なるディレクトリを環境変数 ADMFS に指定してください。また，環境変数 TPFS および環境変数 TPSPOOL と異なる値にしてください。

- ADMSPOOL

  TPBroker の運用支援機能の情報を格納するディレクトリを指定します。
  admsetup コマンドに -c オプションを付けて実行したときに，この環境変数に指定されているディレクトリが生成されます。
  この環境変数が設定されていない場合は，$TPDIR/spool が使用されます。
  同一ノード内に複数の TPBroker の運用支援機能の実行環境を登録する場合は，それぞれの環境で異なるディレクトリを環境変数 ADMSPOOL に指定してください。また，環境変数 TPFS および環境変数 TPSPOOL と異なる値にしてください。

● TPB_TRN_TRACE_PATH

トランザクショントレース情報を格納するディレクトリを指定します。
tssetup コマンドを実行したときに，この環境変数に指定されているディレクトリが生成されます。
この環境変数が設定されていない場合は，環境変数 TPSPOOL に指定したディレクトリが使用されます。

● TPFS

トランザクションのステータスファイルを格納するディレクトリ，またはキャラクタ型スペシャルファイルを指定します。
キャラクタ型スペシャルファイルを指定する場合は，あらかじめ tsmkfs コマンドで TPBroker ファイルシステムを作成しておく必要があります。TPBroker ファイルシステムについては，「7.6　TPBroker ファイルシステム（UNIX）」を参照してください。
この環境変数が設定されていない場合は，環境変数 TPSPOOL に指定したディレクトリが使用されます。
同一ノード内に複数の OTS 実行環境を起動する場合は，それぞれの環境で異なるディレクトリを環境変数 TPFS に指定してください。また，環境変数 ADMFS および環境変数 ADMSPOOL と異なる値にしてください。

● TPRMINFO（C++）

アプリケーションプログラムおよび決着プロセスが使用するリソースマネージャを識別するための定義キーの名称を指定します。設定内容の詳細は「8.3.3　リソースマネージャ定義（C++)」を参照してください。

● TPSPOOL

TPBroker 稼働情報などを格納するディレクトリを指定します。tssetup コマンドを実行したときに，この環境変数に設定されているディレクトリが生成されます。
この環境変数が設定されていない場合は，$TPDIR/otsspool が使用されます。同一ノード内に複数の運用支援機能の実行環境，または複数の OTS 実行環境を起動する場合は，それぞれの環境で異なるディレクトリを環境変数 TPSPOOL に指定してください。また，環境変数 ADMFS および環境変数 ADMSPOOL と異なる値にしてください。

● TPSYS_APTTRCPERTHRD（C++）

～ ((1 ～ 255))<<32>>(単位：トレース)
一つのスレッドで格納できる API トレース数を指定します。この値を超える回数の API が同一スレッドから発行された場合は，古い API トレースから上書きされます。
無効な値を指定した場合は，デフォルト値が適用されます。ただし，tx_info および get_transaction_name オペレーションを発行した場合は，取得情報が通常よりも多くなるため，格納できる API トレース数が指定値よりも少なくなります。

● TPSYS_APTUSE（C++）

API トレースを取得するかどうかを指定します。API トレースを取得するには，任意の値（例えば 1）を指定します。

この環境変数が設定されていない場合は，API トレースは取得されません。

2.3　TPBroker の OTS 環境をセットアップする

tssetup コマンドで TPBroker の環境をセットアップします。

2.3.1　TPBroker の OTS 環境のセットアップ

tssetup コマンドを使用して TPBroker の OTS 環境をセットアップします。tssetup コマンドを実行すると，作業用ディレクトリの作成，環境の初期化，およびシステム環境定義のデフォルト値が設定されます。TPBroker では，システム環境定義設定コマンドを使用してこのデフォルト値を変更できます。ただし，TPBroker の稼働中に定義を変更した場合，その変更は稼働中の TPBroker には反映されません。変更が反映されるのは次回の起動時からです。

tssetup コマンドおよび定義設定コマンドの詳細については，「9.3　運用コマンドの詳細」を参照してください。

環境変数 TPSPOOL，TPFS に設定されたディレクトリ（設定されていなければ $TPDIR/otsspool）を作業用ディレクトリとして生成します。同一ノード内に複数の TPBroker の環境をセットアップする場合は，それぞれの環境で環境変数 TPSPOOL，TPFS に異なる値を設定し，実行環境を分けてください。

tssetup コマンドを実行すると，システム環境定義が初期値に設定されます。バージョンアップなどで再度 tssetup コマンドを実行する場合，前回までのシステム環境定義は上書きされます。このため，tssetup コマンドの実行前に tslsconf コマンドの実行結果をファイルに書き込むなどして，システム環境定義値を保存するようにしてください。

2.4　システム環境定義を変更する

TPBroker では，定義によって実行環境ごとにシステムをカスタマイズできます。必要に応じて，システム環境定義を変更してください。

システム環境定義には，次に示す項目があります。tsdefvalue コマンドを使用して設定します。

- 運用定義
- トランザクション定義
- リソースマネージャ定義（C++）
- 回復定義
- トランザクションコンテキストサーバ定義（Java）
- システム定義
- トランザクショントレース定義

システム環境定義の詳細は，「8.3　システム環境定義の詳細」を参照してください。
tsdefvalue コマンドの詳細は，「9.3　運用コマンドの詳細」を参照してください。

2.5　リソースマネージャと連携する場合の準備（C++）

TPBroker は，DBMS などのリソースマネージャと連携して使用できます。ここでは，XA インタフェースをサポートしたリソースマネージャと連携する場合の準備について，次に示す順に説明します。

1. リソースマネージャを TPBroker に登録する
2. リソースマネージャをシステム環境定義に登録する
3. アプリケーションプログラムとのリンクを設定する

なお，TPBroker に登録されている，またはアプリケーションプログラムとリンクしているリソースマネージャの情報を確認するには，tslsrm コマンドを使用します。このコマンドを実行すると，リソースマネージャ名，スイッチ名，オブジェクト名といったリソースマネージャの情報が標準出力に出力されます。

TPBroker とリソースマネージャとを XA 連携して使用する場合のセットアップについては，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

リソースマネージャとの連携は，Cosminexus TPBroker ではサポートしていません。

2.5.1　リソースマネージャを TPBroker に登録する

リソースマネージャを TPBroker と連携して使う場合，そのリソースマネージャを TPBroker に登録する必要があります。リソースマネージャを登録するには，tslnkrm コマンドを使用します。tslnkrm コマンドの詳細については，「9.3　運用コマンドの詳細」を参照してください。

(1)　リソースマネージャの登録

TPBroker をインストールおよびセットアップしたとき，回復プロセスおよび決着プロセスは，TPBroker が提供する Communication Resource Manager（OTSCRM）の XA インタフェース用のオブジェクトファイルをリンクしています。

TPBroker でリソースマネージャと連携してトランザクションを実行する場合は，トランザクションサービスを開始する前に，TPBroker の tslnkrm コマンドを使用して実行環境にリソースマネージャを登録してください。TPBroker では，最大 7 個のリソースマネージャを登録できます。

(2)　リソースマネージャの削除

TPBroker 下で実行されるトランザクションからアクセスしなくなったリソースマネージャを削除する場合は，tslnkrm コマンドに -d オプションを付けて実行します。

2.5.2　リソースマネージャをシステム環境定義に登録する

リソースマネージャを使用する場合，システム環境定義のリソースマネージャの項目にそのリソースマネージャに関する情報を登録する必要があります。登録する内容には，リソースマネージャ固有の項目もあります。リソースマネージャ固有の項目については，使用するリソースマネージャのマニュアルを参照してください。

また，システム環境定義のリソースマネージャの項目については，「8.3.3　リソースマネージャ定義（C++)」を参照してください。

2.5.3　アプリケーションプログラムとのリンクを設定する

アプリケーションプログラムの実行形式ファイルを作成するときに，トランザクション制御用オブジェクトファイルまたはトランザクション制御用共用ライブラリを作成し，リソースマネージャのライブラリとオブジェクトモジュールをリンクさせる必要があります。リソースマネージャのライブラリとオブジェクトモジュールについては，使用するリソースマネージャのマニュアルを参照してください。

(1)　オブジェクトファイルを作成する

トランザクション制御用オブジェクトファイルは，tsmkobj コマンドを入力して作成します。または tslnkrm コマンドで作成される標準トランザクション制御用オブジェクトファイルを使用することもできます。tsmkobj，tslnkrm コマンドの詳細については，「9.3　運用コマンドの詳細」を参照してください。

(2)　オブジェクトファイルとのリンクを設定する

(a)　すべてのリソースマネージャにアクセスする場合

TPBroker に登録されているすべてのリソースマネージャにアクセスするアプリケーションプログラムの場合は，TPBroker が提供する標準トランザクション制御用共用ライブラリまたは標準トランザクション制御用オブジェクトファイル（$TPSPOOL/XA/tp_allrm.o）をリンクします。

標準トランザクション制御用共用ライブラリを使用する場合は，$TPSPOOL/XA/tp_xaobj.o もアプリケーションプログラムとリンクする必要があります。標準トランザクション制御用共用ライブラリは，tslnkrm コマンドを実行するたびにアップデートされますが，再びアプリケーションプログラムとリンクする必要はありません。

標準トランザクション制御用オブジェクトファイル（tp_allrm.o）は，tslnkrm コマンドを実行するたびにアップデートされるので，リソースマネージャの登録または削除をした場合には，再びアプリケーションプログラムとリンクする必要があります。

(b)　一部のリソースマネージャにアクセスする場合

TPBroker に登録されているリソースマネージャのうち，一部のリソースマネージャだけにアクセスするアプリケーションプログラムの場合は，tsmkobj コマンドで新たな標準

トランザクション制御用共用ライブラリと標準トランザクション制御用オブジェクトファイルを作成して，それらのどちらかとアプリケーションプログラムをリンクします。

(c) リソースマネージャを登録または削除した場合

TPBroker に新たにリソースマネージャを登録した場合，およびリソースマネージャを削除した場合は，そのリソースマネージャを含んだ標準トランザクション制御用オブジェクトファイルとリンクしているアプリケーションプログラムを，再びリンクする必要があります。

標準トランザクション制御用共用ライブラリとリンクしているアプリケーションプログラムは，再びリンクする必要はありません。

(d) リソースマネージャにアクセスしない場合

トランザクション内でリソースマネージャにアクセスしないアプリケーションプログラムの場合，TPBroker が提供している $TPSPOOL/XA/tp_crm.o とリンクする必要があります。このオブジェクトファイルとリンクしていない場合，トランザクションサービスの機能は使用できません。

2.6 プロセス監視定義ファイルを編集する

TPBroker の開始およびセットアップには，プロセス監視用の定義ファイルが必要です。このため，TPBroker をセットアップする前に，任意のテキストエディタでプロセス監視定義ファイルを編集してください。TPBroker では，デフォルトのプロセス監視定義ファイルとして $TPDIR/adm/admconf.cf を提供しています。ただし，$TPDIR/adm/admconf.cf ファイルでは，TCS を自動的に起動する定義部分に「#」を付けてコメントとしています。TCS を自動的に起動する場合，該当する定義部分から「#」を削除してください。

また，Windows Vista 対応版 05-19 以降では，OTS デーモンを起動する設定にも「#」を付けてコメントとしています。OTS デーモンを自動的に起動する場合，該当する定義部分から「#」を削除してください。

プロセス監視定義ファイルには，次に示す項目を設定します。

- 監視対象プロセスの識別子（必須）
- 監視対象プロセスファイル名（必須）
- プロセス起動用コマンドのタイムアウト値（任意）
- プロセス起動に失敗した場合の TPBroker の処置（必須）
- 監視対象プロセスが異常終了した場合の TPBroker の処置（必須）
- プロセス起動のタイミングの指定（必須）
- 一定時間内に連続して異常終了する回数の上限値（必須）
- 正常停止用コマンド（任意）
- 強制停止用コマンド（任意）
- コマンド実行ユーザ ID（任意）
- コマンド実行グループ ID（任意）
- プロセス起動時に指定する環境変数名と値（任意）
- TPBroker 開始時に指定する環境変数名と値（任意）

なお，プロセス監視と定義項目の詳細については，「6.2　プロセス監視」を参照してください。

2.7 TPBroker の運用支援機能実行環境をセットアップする

admsetup コマンドで TPBroker の運用支援機能実行環境をセットアップします。

2.7.1 TPBroker の運用支援機能実行環境の初期化

admsetup コマンドを実行すると，次に示す環境変数に指定した作業用ディレクトリが自動的に生成されます。

● ADMSPOOL
トレースファイル，メモリマップドファイルを格納するディレクトリが指定されています。

● ADMFS
システムステータスファイルを格納するディレクトリが指定されています。

なお，TPBroker の OTS 環境を使用しない場合でも，admsetup コマンドの実行前に tssetup コマンドで OTS 環境をセットアップしてください。

2.7.2 TPBroker の運用支援機能実行環境の OS への登録

OS が起動するときに TPBroker の運用支援機能実行環境が自動的に起動されるようにするために，TPBroker を OS へ登録します。

admsetup コマンドを実行すると，TPBroker は OS に登録されます。TPBroker の OS への登録を解除するには，admsetup コマンドに -d オプションを付けて実行します。

(1) UNIX 版の場合

admsetup コマンドでセットアップを実行すると，TPBroker は /etc/inittab に登録されます。以降，OS を起動すると自動的に TPBroker が開始されます。一つの OS に複数の TPBroker を登録できます。

一つの OS に複数の TPBroker を登録する場合は，それぞれの TPBroker で環境変数 TPSPOOL，ADMSPOOL，および ADMFS が異なるように設定してください。

(2) Windows 版の場合

admsetup コマンドでセットアップを実行すると，TPBroker は Windows のサービスとして登録されます。この場合，サービス名称は「TPBroker」となります。TPBroker 05-15 以降では，指定したサービス名で Windows のサービスとして登録されます。また，TPBroker 05-15 以降では一つの OS に複数の TPBroker を登録できます。

TPBroker 05-15 以降では，指定したサービス名でサービスとして登録されますが，以降の説明では，サービス「TPBroker」と記述します。

(a) サービスコントロールパネルで TPBroker を開始する場合

サービス「TPBroker」を開始すると，TPBroker が開始されて使用できるようになります。運用定義 /ADM/set_conf_mode の設定が "AUTO" の場合，自動的にオンライン状態（プロセス監視開始）になります。運用定義については，「8.3.1　運用定義」を参照してください。

サービスコントロールパネルからサービス「TPBroker」を開始する場合，デフォルトで指定された運用支援機能で使用するプロセス監視定義ファイルを基に，ADM デーモンが起動されます。デフォルトのプロセス監視定義ファイルは admsetup コマンド実行時に引数で指定した定義ファイルです。

また，サービスコントロールパネルからサービス「TPBroker」を開始する場合，スタートアップパラメタにプロセス監視定義ファイル名を指定すると，ADM デーモン起動時に使用するプロセス監視定義ファイルを変更できます。プロセス監視定義ファイル名は絶対パス名で指定してください。このとき，スタートアップパラメタで指定されたプロセス監視定義ファイル名は，次回 ADM デーモンがサービス「TPBroker」から起動されるときのデフォルトのプロセス監視定義ファイル名になります。

(b) サービスコントロールパネルで TPBroker を終了する場合

サービス「TPBroker」を停止すると，TPBroker は強制正常終了します。このとき，運用定義 /ADM/service_stop_mode の設定値に応じて admstop コマンドが -f または -fr オプション付きで自動的に発行され，ADM デーモンが停止します。また，マシンのシャットダウン時は TPBroker は強制正常終了します。

(c) 注意事項

● マシンをシャットダウンした場合
マシンをシャットダウンした場合，TPBroker のデーモンプロセスや TPBroker の監視対象プロセスも，OS によって強制停止されます。

● サービスの対話許可の設定
サービス「TPBroker」のスタートアップパラメタのデフォルトは,「デスクトップとの対話をサービスに許可」のチェックボックスがオフになっています。このため，TPBroker から起動および監視されるプロセスはデスクトップには表示されないので，ダイアログなどをデスクトップに表示するようなプロセスでは，それらのダイアログも表示されません。
監視対象プロセスがダイアログなどを表示する場合は，Windows のサービスの設定で「デスクトップとの対話をサービスに許可」チェックボックスをオンにしてから，TPBroker を開始してください。
ただし，TPBroker の稼働中にこの設定でログオフすると，監視対象プロセスが Windows の GUI を持つアプリケーションプログラムだった場合，このアプリケー

ションプログラムは異常終了することがあります。必ず，そのアプリケーションプログラムを監視対象から外すか，またはサービス「TPBroker」を停止してから，ログオフしてください。

3 OTS 機能

TPBroker の OTS には，C++ 実行環境および Java 実行環境で使用できる OTS と，C++ 実行環境だけで使用できる C++ OTS と，Java 実行環境だけで使用できる Java OTS があります。この章では，C++ 実行環境および Java 実行環境で使用できる OTS の機能について説明します。
C++ 実行環境だけで使用できる C++ OTS については，「4. C++ OTS 機能（C++）」を参照してください。
Java 実行環境だけで使用できる Java OTS については，「5. Java OTS 機能（Java）」を参照してください。

3.1　トランザクション制御

TPBroker が提供する OTS 機能では，CORBA で規定された OTS の仕様に基づいてトランザクションを制御します。ここでは，CORBA で規定された OTS の主な仕様，および TPBroker の OTS 機能について説明します。Cosminexus TPBroker は，ここで説明している OTS 機能をベースに，J2EE サーバ上での分散トランザクション処理を実現するための機能を提供しています。

3.1.1　CORBA で規定された OTS の仕様

OTS は，CORBA システム上のオブジェクトのオペレーション起動に ACID 特性を与えるための機能を提供します。ACID 特性とは，原子性（Atomicity），一貫性（Consistency），分離性（Isolation），および持続性（Durability）の四つの特性です。クライアントが OTS へトランザクションの開始を依頼してから，OTS へトランザクションの終了を依頼するまでのリソースのアップデートを伴うすべてのリクエスト処理は，ACID 特性が保証されます。

また，OTS は各グローバルトランザクションの状態を表すトランザクションコンテキストを管理します。アプリケーションプログラムは，Terminator オブジェクト，Coordinator オブジェクトなど，OTS が提供する CORBA システム上のオブジェクトにリクエストを発行し，トランザクションを制御します。これらのオブジェクトは，グローバルトランザクションを単位とするトランザクションコンテキストを共有します。

(1)　コミットとロールバック

トランザクションの終了はコミットまたはロールバックのどちらかに必ず決められます。コミットは，トランザクション中で起きたすべての変更を永久的なものとします。ロールバックは，トランザクション中に起きたすべての変更を取り消します。

(2)　トランザクショナルアプリケーション

トランザクションを生成したアプリケーションプログラムは，トランザクションに含まれ，トランザクショナルアプリケーションと呼ばれます。トランザクショナルアプリケーションから CORBA オブジェクトにアクセスした場合，呼び出した CORBA オブジェクトをグローバルトランザクションに含めることができます（トランザクショナルオペレーション）。さらに，呼び出した CORBA オブジェクトから別の CORBA オブジェクトを呼び出すこともできます。

また，リソースを含んだオブジェクトがトランザクショナルアプリケーションに含まれる場合，リソースの状態はトランザクションと同期を取ります。

(3)　トランザクショナルクライアント

トランザクショナルクライアントは，一つのトランザクション内で複数のトランザク

ショナルオブジェクトのオペレーションを起動できるプログラムです。トランザクショナルオブジェクトのうち，トランザクションを開始する任意のプログラムを，トランザクションオリジネータといいます。

(4) トランザクショナルオブジェクト

トランザクション内で起動されると，その動作に影響を及ぼされるオブジェクトです。トランザクショナルオブジェクトは，トランザクション中で変更される永続的なデータを含んでも，含まなくてもかまいません。トランザクション内でリクエストが発行されていても，トランザクションサービスには，すべてのリクエストがトランザクショナルな動作を備えている必要はありません。また，リクエストに対してトランザクショナルな動作をするかどうかを選択することもできます。

トランザクショナルオブジェクトは，トランザクショナルサーバとリカバラブルサーバを実装するのに使用されます。

(5) リカバラブルオブジェクトとリソースオブジェクト

リカバラブルオブジェクトは，トランザクショナルオブジェクトの一種でトランザクションのスコープで変更されたデータを直接管理します。リカバラブルオブジェクトは，トランザクションのコミットまたはロールバックと同期を取るデータを持ちます。

コミットまたはロールバックに参加するために，リカバラブルオブジェクトは，トランザクションサービスにリソースオブジェクトを登録します。トランザクションサービスは，登録されたリソースオブジェクトへコミットまたはロールバックと同期を取るように指示を出します。

(6) トランザクショナルサーバ

トランザクショナルサーバは一つ以上のトランザクショナルオブジェクトの集まりで，リカバラブルオブジェクトではありません。それ自身では回復可能な状態を持ちません。トランザクションの終了プロセスには参加しませんが，トランザクションをロールバックすることはできます。

(7) リカバラブルサーバ

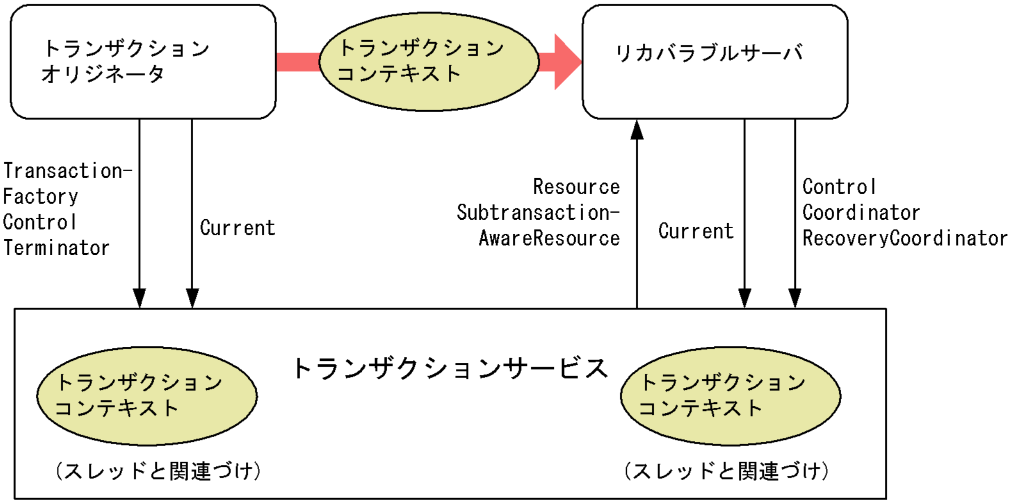
リカバラブルサーバはオブジェクトの集まりであり，少なくともその一つが回復できます。

3.1.2 トランザクション制御の概要

TPBroker はトランザクションやプロセスごとにオブジェクトを持ち，そのオブジェクトインタフェースを使用してトランザクション制御をします。

TPBroker の主要なコンポーネントとオブジェクトインタフェースについて，次の図に示します。

図 3-1　TPBroker の主要なコンポーネントおよびオブジェクトインタフェース



(1)　コンポーネント

(a)　トランザクションオリジネータ

トランザクションオリジネータは，トランザクションを開始する任意のプログラムです。リカバラブルサーバは，トランザクション内でトランザクションオリジネータによって直接的に，または一つ以上のトランザクショナルオブジェクトを介して間接的に起動される，回復可能な状態を持つオブジェクトを実装します。

(b)　リカバラブルサーバ

リカバラブルサーバは，Resource を Coordinator に登録します。Resource は，トランザクションサービスによって起動される 2 相コミットプロトコルを実装しています。

(2)　オブジェクトインタフェース

TPBroker は次に示すオブジェクトインタフェースを提供しています。各オブジェクトインタフェースについては，「3.1.3　オブジェクトトランザクションサービス」を参照してください。

- Control
- Coordinator
- Current
- RecoveryCoordinator
- Resource
- SubtransactionAwareResource
- Synchronization
- Terminator
- TransactionalObject

- TransactionFactory

3.1.3 オブジェクトトランザクションサービス

TPBroker は，OMG の OTS の仕様に準拠した機能を提供しています。

オブジェクトトランザクションは，TPBroker が提供するオブジェクトと，リソースベンダまたはユーザが実装するオブジェクトに分けられます。

(1) TPBroker が提供するオブジェクトインタフェース

(a) Control

Control は，プログラムがトランザクションコンテキストを明示的に管理したり，プロパゲーションしたりするためのインタフェースです。Terminator と Coordinator を得るためのオペレーションを提供します。このインタフェースをサポートしているオブジェクトは，トランザクションごとに存在します。

(b) Coordinator

Coordinator は，明示的にまたは暗黙的に（リクエストとともにトランザクションコンテキストをプロパゲーションすると）リカバラブルサーバを利用できるようにします。

Coordinator は，トランザクションごとに存在し，そのトランザクションに対するトランザクション情報の取得，サブトランザクションの生成，およびリソースの登録など，各種オペレーションを提供します。

(c) Current

Current は，トランザクションの開始・終了と現在のトランザクションの情報を取得します。プロセスごとに存在します。

プロセス中の各スレッドに対応したトランザクションコンテキストを保持し，動作中のスレッドに対応したトランザクションコンテキストによって制御されます。アプリケーションプログラムは，この Current が提供するオペレーションによってコンテキストの管理操作ができます。

コーディングを単純化するために，ほとんどのアプリケーションプログラムは，暗黙的なスレッドごとに，トランザクションコンテキストへのアクセスを提供する Current オブジェクトを使用します。

(d) RecoveryCoordinator

RecoveryCoordinator は，register_resource オペレーションで登録された Resource オブジェクトに対して，障害時の回復用のオペレーションを提供します。トランザクションごとに存在します。

(e) Terminator

Terminator は，トランザクションを終了させるためのオペレーションを提供します。トランザクションオリジネータは，Terminator インタフェースを使用してトランザクションのコミットおよびロールバックをします。

(f) TransactionFactory

TransactionFactory は，トップトランザクションを生成するオペレーションを提供します。また，Terminator や Coordinator へのアクセスを提供する Control が戻り値として返されます。

(2) リソースベンダまたはユーザが実装するオブジェクトインタフェース

(a) Resource

Resource は，登録されたリソース上でトランザクションサービスによって起動されるオペレーションを提供します。Resource はある障害が発生したときに，トランザクションの結果を判断するため，およびトランザクションサービスと回復処理を協調させるために RecoveryCoordinator を使用します。

(b) SubtransactionAwareResource

SubtransactionAwareResource は，Resource インタフェースに加えて，サブトランザクションの完了の通知を受けるためのインタフェースです。Coordinator の register_subtran_aware オペレーションでサブトランザクションに対して登録できます。ただし，この登録はサブトランザクションの完了の通知を受けられるだけです。トップトランザクションの完了処理に参加する場合は，別途 Coordinator の register_resource オペレーションでトップトランザクションに対して登録する必要があります。

(c) Synchronization

Synchronization は，2 相コミットプロトコルの開始前と完了後に通知を受けるためのインタフェースを持ちます。Coordinator の register_synchronization オペレーションでトップトランザクションに対して登録できます。

(d) TransactionalObject

TransactionalObject は，OTS 1.1 との互換性を保つためだけに残されたインタフェースです。OTS 1.3 に準拠したアプリケーションプログラムが，OTS 1.1 に準拠したアプリケーションプログラムとトランザクション連携する場合にだけ必要となります。

OTS 1.1 に準拠したアプリケーションプログラムと連携する場合，TransactionalObject は，このインタフェースを実装したオブジェクトがトランザクショナルであることを示し，そのオブジェクトに対するオペレーション実行時にトランザクションが暗黙的にプロパゲーションされることを意味します。

3.1.4 トランザクションモデル

OTS のトランザクションモデルには，フラットトランザクションとネスティッドトランザクションがあります。

(1) フラットトランザクション

フラットトランザクションとは，「3.1.1 CORBA で規定された OTS の仕様」で説明した ACID 特性を持つトランザクションのことであり，X/Open DTP モデルのトランザクションと互換性があります。ネスティッドトランザクションに比べて単純なモデルであり，すべてのトランザクション処理を実現できるため，一般的なアプリケーションプログラムではこのフラットトランザクションを使用してください。

(2) ネスティッドトランザクション

ネスティッドトランザクションとは，トランザクションをネストさせることができるモデルです。トランザクションをネストさせると，部分的なロールバックができます。一つの大きなトランザクション処理を幾つかの処理単位に分割し，ある処理単位が失敗した場合に，その処理単位だけを部分的にロールバックして代替処理を行い，全体のトランザクションをコミットさせます。このため，部分的な障害をトランザクション全体に影響させないようにできます。

このようにネスティッドトランザクションは，高度に分散された単一のトランザクション処理をするシステムで，システム全体のスループットを向上させることができます。

なお，C++ 実行環境で XA インタフェースおよび TX インタフェースを使用する場合は，このネスティッドトランザクションを使用しないでください。

- サブトランザクション
  ネスティッドトランザクションでは，あるトランザクションの処理中にサブトランザクションと呼ばれる子トランザクションを生成できます。また，そのサブトランザクションの処理中には，さらにその子となるサブトランザクションを生成できます。
- 親トランザクションと子トランザクション
  あるトランザクション中にサブトランザクションを生成したとき，元のトランザクションをそのサブトランザクションに対する親トランザクションと呼び，生成されたサブトランザクションを元の親トランザクションの子トランザクションと呼びます。一つの親トランザクションが複数の子トランザクションを持つことができ，TPBroker での一つの親トランザクションが持つことのできる子トランザクションの最大数は 256 です。トランザクションのネストのレベル（深さ）は最大 32 レベルです。
- トランザクションファミリー
  あるサブトランザクションから，その親を含む，再帰的なすべての親を先祖（ancestor）と呼び，あるトランザクションから，その子を含む，すべての再帰的な子を子孫（descendant）と呼びます。親を持たない最上位のトランザクションのことをトップトランザクションと呼び，トップトランザクションとそのすべての子孫のことをトランザクションファミリーと呼びます。

サブトランザクションがロールバックすると，先祖のトランザクションの意思に関係なく，そのサブトランザクションで行った処理はロールバックされます。サブトランザクションがコミットした場合，その時点では実際にコミットするかロールバックするかは決定されないで，そのサブトランザクションの先祖の意思にゆだねられます。すべての先祖がコミットすればそのサブトランザクションもコミットされますが，先祖のうち一つでもロールバックした場合はそのサブトランザクションもロールバックされます。したがって，あるトランザクションが完了（コミットまたはロールバック）するには，そのすべての子トランザクションが完了していなければなりません。TPBroker は，トランザクションのコミット要求時にすべての子トランザクションが完了していない場合は，そのトランザクションをロールバックさせます。

3.1.5 コンテキスト管理

アプリケーションプログラムがトランザクションコンテキストを管理する方式として間接コンテキスト管理と直接コンテキスト管理があります。一般的なアプリケーションプログラムでは間接コンテキスト管理を使用してください。アプリケーションプログラムが特別なトランザクションコンテキスト操作を行う場合だけ，直接コンテキスト管理を使用してください。

(1) 間接コンテキスト管理

アプリケーションプログラムが Current 擬似オブジェクトを使用して，アプリケーションプログラムのスレッドとトランザクションコンテキストの関連づけをします。

(2) 直接コンテキスト管理

アプリケーションプログラムが Current 擬似オブジェクトを使用したトランザクションの関連づけをしないで，管理したいトランザクションの Control や Coordinator などのオブジェクトを直接操作してトランザクションの制御をします。

3.1.6 プロパゲーション

クライアントアプリケーションがサーバオブジェクトに対してオペレーション実行時に，クライアント側のトランザクションコンテキストをサーバに伝達できます。これをプロパゲーションといいます。プロパゲーションには，暗黙的プロパゲーションと明示的プロパゲーションの 2 種類があります。一般的なアプリケーションプログラムでは，暗黙的プロパゲーションを使用します。直接コンテキスト管理を使用している場合や，特別なプロパゲーションルールを設けたい場合だけ，明示的プロパゲーションを使用してください。

(1) 暗黙的プロパゲーション

サーバオブジェクトへのリクエスト発行時に，間接コンテキスト管理によってクライアント側の Current 擬似オブジェクトと関連づけられているトランザクションコンテキストを，トランザクションサービスがリクエストとともに暗黙的にプロパゲーションしま

す。このため，クライアントがトランザクションコンテキストに直接干渉することなくプロパゲーションが行われます。暗黙的プロパゲーションを使用する場合は，サーバ側 POA ポリシーの OTSPolicy に ADAPTS，または REQUIRES を設定していなければなりません。直接コンテキスト管理を行っている場合，またはサーバ側 POA ポリシーの OTSPolicy に FORBIDS を設定している場合は，リクエスト時にトランザクションサービスによる暗黙的プロパゲーションは行われません。

(a)　OTS 1.1 準拠のアプリケーションプログラムとの相互接続

暗黙的プロパゲーションでは，OTS 1.1 準拠のアプリケーションプログラム（TPBroker Version 3 のアプリケーションプログラムなど）と相互接続できる場合があります。暗黙的プロパゲーションでの，OTS 1.1 準拠のアプリケーションプログラムとの相互接続性を次の表に示します。

表 3-1　OTS 1.1 準拠のアプリケーションプログラムとの相互接続性（暗黙的プロパゲーション）

クライアント		サーバ			
		TPBroker Version 3		TPBroker Version 5	
		C++	Java	C++	Java
TPBroker Version 3	C++	○	○	×	×
	Java	○	○	×	×
TPBroker Version 5	C++	○※	○	○	○
	Java	○※	○	○	○

（凡例）
　○：接続できる
　×：接続できない

注※
　TPBroker Version 5 のクライアントアプリケーションは，TPBroker Version 3 C++ のシングルスレッドのサーバとは接続できません。TPBroker Version 3 C++ のサーバと接続するには，TPBroker Version 3 C++ のサーバをマルチスレッドにしてください。

(2)　明示的プロパゲーション

トランザクションサービスによって定義された Control や Coordinator などのオブジェクトのリファレンスをパラメタとして明示的にサーバオブジェクトに渡すことによって，アプリケーションプログラムがトランザクションコンテキストをプロパゲーションします。サーバ側 POA ポリシーの OTSPolicy には，通常は FORBIDS を設定します。明示的プロパゲーションを使用すると，チェックドトランザクション機能が無効となりますので注意してください。

(a) OTS 1.1 準拠のアプリケーションプログラムとの相互接続性

明示的プロパゲーションでは，OTS 1.1 準拠のアプリケーションプログラム（TPBroker Version 3 のアプリケーションプログラムなど）と相互接続できます。明示的プロパゲーションでの，OTS 1.1 準拠のアプリケーションプログラムとの相互接続性を次の表に示します。

表 3-2　OTS 1.1 準拠のアプリケーションプログラムとの相互接続性（明示的プロパゲーション）

クライアント		サーバ			
		TPBroker Version 3		TPBroker Version 5	
		C++	Java	C++	Java
TPBroker Version 3	C++	○	○	○	○
	Java	○	○	○	○※
TPBroker Version 5	C++	○	○	○	○
	Java	○	○	○	○

（凡例）
○：接続できる

注※
明示的プロパゲーションで呼び出された TPBroker Version 3 のクライアントアプリケーションが，TPBroker Version 5 のサーバを呼び出すと，INVALID_TRANSACTION 例外が返されることがあります。これを回避するには，TPBroker Version 3 のクライアントアプリケーションを次のように修正してください。

- TPBroker Version 5 のサーバを呼び出す前に，Current オブジェクトの suspend オペレーションを実行し，スレッドとトランザクションとの関連づけをなくす。
- TPBroker Version 5 サーバの呼び出し後に，Current オブジェクトの resume オペレーションを実行し，再度スレッドとトランザクションとを関連づける。

3.1.7 チェックドトランザクション

TPBroker では，トランザクションの完全性を保証するために，次に示すチェック機能をサポートします。これによって OTS の機能の使用方法に制限が加えられます。ただし，次に示すチェック機能が有効となるのは，暗黙的プロパゲーションを使用する場合だけです。明示的プロパゲーションを使用する場合は，アプリケーションプログラム側でトランザクションの完全性を確保してください。

(1) 応答チェック

トランザクショナルなリクエストからクライアントに応答を返す前に，すべての遅延同

期リクエストの応答が受信されているかどうかをチェックします。同期リクエストに関しては，応答が返ってこないと処理が続行されないため，暗黙的に応答が保証されます。チェックの結果，応答を受信していないリクエストが存在する場合，CORBA::TRANSACTION_ROLLEDBACK 例外が発生し，そのトランザクションをロールバックします。

(2) リジュームチェック

アプリケーションプログラムが，Current オブジェクトの resume オペレーションによってトランザクションコンテキストをスレッドに関連づける前に，このトランザクションコンテキストがそのスレッドで生成されたものか，またはそのスレッドに暗黙的にプロパゲーションされたものかどうかをチェックします。チェックの結果，トランザクションコンテキストが無効な場合，CosTransactions::InvalidControl 例外が発生します。

(3) コミットチェック

アプリケーションプログラムからのコミット要求時，コミット処理の開始前に，次に示すチェックをします。

1. トランザクションが生成されたスレッドと同じスレッドからコミット要求が発行されているかどうか。
2. コミット要求が出されているクライアントが，すべての遅延同期リクエストに対する応答を受信済みかどうか。

1. のチェック条件を満たしていない場合，CORBA::INVALID_TRANSACTION 例外が発生します。2. のチェック条件を満たしていない場合，CORBA::TRANSACTION_ROLLEDBACK 例外が発生します。また，1. の条件を満たしていない場合で，直接コンテキスト管理を使用するときには，Control オブジェクトの get_terminator オペレーションで CosTransactions::Unavailable 例外が発生するため，コミット要求できません。

3.1.8 ポリシーの設定

サーバオブジェクト，またはクライアントアプリケーションに，OTS が提供するポリシーを設定すると，トランザクションコンテキストの暗黙的プロパゲーションを制御できます。

(1) OTS が提供するポリシー

OTS が提供するポリシーには，サーバ側で設定するポリシー（OTSPolicy, InvocationPolicy）と，クライアント側で設定するポリシー（NonTxTargetPolicy）があります。

OTSPolicy

OTSPolicy は，サーバ側で設定するポリシーです。OTSPolicy を設定すると，サー

バは，トランザクションコンテキストの暗黙的プロパゲーションを必要とするかどうかを，クライアントに宣言できます。OTSPolicy を設定していないサーバは，値が FORBIDS（デフォルト値）の OTSPolicy が設定された場合と同じです。

OTSPolicy には，ADAPTS，FORBIDS，および REQUIRES の三つの値があります。値の意味を次の表に示します。

表 3-3　OTSPolicy の値の意味

OTSPolicy の値	意味
ADAPTS	この値の OTSPolicy を設定したサーバは，間接コンテキスト管理を使用してトランザクション処理を開始しているクライアント，および間接コンテキスト管理を使用しないでトランザクション処理を開始しているクライアントのどちらでも呼び出せます。前者のクライアントは，サーバにトランザクションコンテキストを暗黙的にプロパゲーションできます。後者のクライアントは，サーバに通常の呼び出し※をすることができます。
FORBIDS（デフォルト値）	この値の OTSPolicy を設定したサーバは，トランザクションコンテキストの暗黙的プロパゲーションを受け入れません。間接コンテキスト管理を使用しないでトランザクション処理を開始しているクライアントの場合，クライアントはサーバに通常の呼び出し※をすることができます。間接コンテキスト管理を使用してトランザクション処理を開始しているクライアントの場合，クライアントに設定されている NonTxTargetPolicy の値によって，次に示すどちらかの動作をします。 • INVALID_TRANSACTION 例外が発生し，サーバを呼び出せません。 • 通常の呼び出し※を行います。
REQUIRES	この値の OTSPolicy を設定したサーバは，トランザクションコンテキストの暗黙的プロパゲーションが必要です。間接コンテキスト管理を使用してトランザクション処理を開始しているクライアントの場合，クライアントはサーバにトランザクションコンテキストを暗黙的にプロパゲーションします。間接コンテキスト管理を使用しないでトランザクション処理を開始しているクライアントの場合，クライアントがサーバを呼び出したときに，TRANSACTION_REQUIRED 例外が発生し，サーバを呼び出せません。

注※

トランザクションコンテキストを暗黙的にプロパゲーションしない呼び出しです。

InvocationPolicy

InvocationPolicy は，サーバ側で設定するポリシーです。InvocationPolicy を設定すると，サーバは，中間ルータを介した呼び出しを必要とするかどうかを，クライアントに宣言できます。InvocationPolicy を設定していないサーバは，値が EITHER（デフォルト値）の InvocationPolicy が設定された場合と同じです。

InvocationPolicy には，EITHER，SHARED，および UNSHARED の三つの値があります。値の意味を次の表に示します。

注

TPBroker Version 5 は，中間ルータを介したサーバ呼び出しをサポートしていません。したがって，TPBroker Version 5 で作成するサーバには，

InvocationPolicy を設定する必要はありません。

表 3-4　InvocationPolicy の値の意味

InvocationPolicy の値	意味
EITHER （デフォルト値）	この値の InvocationPolicy を設定したサーバは，中間ルータを介した呼び出し，および介さない呼び出しのどちらでも呼び出せます。
SHARED	この値の InvocationPolicy を設定したサーバは，中間ルータを介さない呼び出しが必要です。中間ルータを介した呼び出しを行うと，サーバ呼び出し時に TRANSACTION_MODE 例外が発生し，サーバを呼び出せません。
UNSHARED	この値の InvocationPolicy を設定したサーバは，中間ルータを介した呼び出しが必要です。中間ルータを介さない呼び出しを行うと，サーバ呼び出し時に TRANSACTION_MODE 例外が発生し，サーバを呼び出せません。

NonTxTargetPolicy

NonTxTargetPolicy は，クライアント側で設定するポリシーです。

NonTxTargetPolicy は，次に示す条件を満たした状況でサーバを呼び出した場合に有効になります。

- クライアントが，サーバ呼び出し前に，間接コンテキスト管理を使用してトランザクション処理を開始している。
- サーバに，値が FORBIDS の OTSPolicy が設定されているか，または OTSPolicy が設定されていない。

NonTxTargetPolicy を設定すると，クライアントは，サーバの OTSPolicy（値は FORBIDS）に従って，トランザクションコンテキストを暗黙的にプロパゲーションする呼び出しから，通常の呼び出し（トランザクションコンテキストを暗黙的にプロパゲーションしない呼び出し）に切り替えるか，切り替えないかを決定できます。NonTxTargetPolicy を設定していないクライアントは，値が PERMIT（デフォルト値）の NonTxTargetPolicy が設定された場合と同じです。

NonTxTargetPolicy には，PERMIT，および PREVENT の二つの値があります。値の意味を次の表に示します。

表 3-5　NonTxTargetPolicy の値の意味

NonTxTargetPolicy の値	意味
PERMIT （デフォルト値）	この値の NonTxTargetPolicy を設定したクライアントは，FORBIDS の OTSPolicy が設定されているサーバに対し，呼び出し方法を，自動的に通常の呼び出し※に切り替えます。間接コンテキスト管理を使用してトランザクション処理を開始しているクライアントは，通常の呼び出しに切り替えられ，サーバを呼び出せます。 この呼び出しでは，トランザクションコンテキストが暗黙的にプロパゲーションされないことに注意してください。

NonTxTargetPolicy の値	意味
PREVENT	この値の NonTxTargetPolicy を設定したクライアントは，FORBIDS の OTSPolicy が設定されているサーバに対し，呼び出し方法を，通常の呼び出し※に切り替えません。間接コンテキスト管理を使用してトランザクション処理を開始しているクライアントが，サーバを呼び出すと，INVALID_TRANSACTION 例外が発生します。 この値を設定すると，クライアントは，トランザクションコンテキストの暗黙的プロパゲーションができないサーバをチェックすることができます。

注※

トランザクションコンテキストを暗黙的にプロパゲーションしない呼び出しです。

(2) ポリシーとトランザクションコンテキストの暗黙的プロパゲーションとの関係

(1) で説明したように，OTSPolicy および NonTxTargetPolicy を設定すると，サーバ呼び出し時にトランザクションコンテキストの暗黙的プロパゲーションを制御できます。

ポリシーの設定と，サーバ呼び出し時のトランザクションコンテキストの暗黙的プロパゲーションの可否の対応を，次の二つの表に示します。

表 3-6 暗黙的プロパゲーションの可否：間接コンテキスト管理を使用してトランザクション処理を開始しているクライアントによる呼び出しの場合

NonTxTargetPolicy の値	OTSPolicy の値		
	ADAPTS	FORBIDS（デフォルト）	REQUIRES
PERMIT （デフォルト）	○	△	○
PREVENT	○	× （INVALID_TRANSACTION 例外）	○

（凡例）

○：トランザクションコンテキストを暗黙的にプロパゲーションする
△：トランザクションコンテキストを暗黙的にプロパゲーションしない
×：サーバを呼び出せない

表 3-7　暗黙的プロパゲーションの可否：間接コンテキスト管理を使用しないでトランザクション処理を開始しているクライアントによる呼び出しの場合

NonTxTargetPolicy の値	OTSPolicy の値		
	ADAPTS	FORBIDS（デフォルト）	REQUIRES
PERMIT（デフォルト）	△	△	×（TRANSACTION_REQUIRED 例外）
PREVENT	△	△	×（TRANSACTION_REQUIRED 例外）

（凡例）
　△：トランザクションコンテキストを暗黙的にプロパゲーションしない
　×：サーバを呼び出せない

間接コンテキスト管理を使用してトランザクション処理を開始しているクライアントは，ADAPTS，または REQUIRES の OTSPolicy が設定されたサーバに，トランザクションコンテキストを暗黙的にプロパゲーションできます。間接コンテキスト管理を使用しないでトランザクション処理を開始しているクライアントは，ADAPTS，または FORBIDS の OTSPolicy が設定されたサーバに，通常の呼び出し（トランザクションコンテキストを暗黙的にプロパゲーションしない呼び出し）を行うことができます。

(3) トランザクション処理に必要なポリシーの設定

トランザクション処理を行うためには，OTSPolicy，NonTxTargetPolicy を設定する必要があります。

(a) 暗黙的プロパゲーションによるトランザクション処理を行う場合

サーバ側の設定（OTSPolicy の設定）

暗黙的プロパゲーションによるトランザクション処理を行う場合，サーバに値が ADAPTS または REQUIRES の OTSPolicy を設定します。
サーバに ADAPTS の OTSPolicy を設定した場合，間接コンテキスト管理を使用しないでトランザクション処理を開始しているクライアントも，サーバを呼び出せます。REQUIRES の OTSPolicy を設定した場合，間接コンテキスト管理を使用してトランザクション処理を開始しているクライアントだけがサーバを呼び出せます。
間接コンテキスト管理を使用しないでトランザクション処理を開始しているクライアントも受け付けたいサーバには，ADAPTS の OTSPolicy を設定してください。間接コンテキスト管理を使用してトランザクション処理を開始しているクライアントだけを受け付けたいサーバには，REQUIRES の OTSPolicy を設定してください。

クライアント側の設定（NonTxTargetPolicy の設定）

クライアントが呼び出すすべてのサーバに，ADAPTS または REQUIRES の

OTSPolicy を設定している場合，NonTxTargetPolicy を設定する必要はありません。しかし，クライアントが呼び出すサーバの中に，FORBIDS の OTSPolicy を設定しているサーバが存在する可能性がある場合，NonTxTargetPolicy の設定について考慮する必要があります。
トランザクションコンテキストの暗黙的プロパゲーションによる呼び出しだけを行いたいクライアントは，FORBIDS の OTSPolicy を設定しているサーバを呼び出さないよう，PREVENT の NonTxTargetPolicy を設定してください。FORBIDS の OTSPolicy を設定しているサーバに対する呼び出しを，自動的に通常の呼び出しに切り替えたい場合，PERMIT の NonTxTargetPolicy を設定してください。

(b) 明示的プロパゲーションによるトランザクション処理を行う場合

明示的プロパゲーションによるトランザクション処理を行う場合，サーバに FORBIDS（デフォルト）の OTSPolicy を設定してください。また，クライアントに PERMIT（デフォルト）の NonTxTargetPolicy を設定してください。

(4) 設定可能なポリシーの組み合わせ

OTSPolicy と InvocationPolicy の両方を設定する場合，値の組み合わせによって，設定できるものとできないものがあります。OTSPolicy と InvocationPolicy の組み合わせを次の表に示します。

表 3-8　設定可能な OTSPolicy と InvocationPolicy の値の組み合わせ

OTSPolicy の値	InvocationPolicy の値			
	EITHER	SHARED	UNSHARED	設定なし
ADAPTS	×	○	×	○
FORBIDS	×	○	×	○
REQUIRES	○	○	○	○
設定なし	×	○	×	○

（凡例）
○：設定できる
×：設定できない

値が SHARED，または設定されていない InvocationPolicy の場合，すべての値の OTSPolicy が設定できます。値が EITHER または UNSHARED の InvocationPolicy の場合，REQUIRES の OTSPolicy だけ設定できます。設定できない組み合わせにすると，InvalidPolicy 例外が発生します。

3.1.9 トランザクション稼働統計情報

TPBroker が起動してからのトランザクション稼働統計情報を出力できます。次に示す情

報を標準出力に出力します。

- TPBroker の OTS 機能の起動時間
- タイムアウトしたトランザクションブランチの数
- トップトランザクションブランチのルートブランチ情報
- サブトランザクションブランチのルートブランチ情報
- サブオーディネートのトランザクションブランチの累計
- 稼働中のトランザクションブランチの数
- コミットしたトランザクションブランチの数
- ロールバックしたトランザクションブランチの数
- ヒューリスティックに決着したトランザクションブランチの数
- リードオンリーのトランザクションブランチの数

3.1.10 トランザクショントレース

トランザクショントレースは，トランザクションブランチの結果を残します。トランザクションブランチが開始，および決着したときに，時刻，決着の結果などの情報をファイルに記録します。

トランザクショントレースを参考にして，トランザクションブランチがヒューリスティックに決着していればデータベースを回復し，ロールバックしていればトランザクションを再開始するなどの判断ができます。また，決着しないで残っているトランザクションブランチがあれば，同じグローバルトランザクション内のトランザクションブランチがどのように決着したかを参考に，tscommit または tsrollback コマンドで決着処理を指示できます。

トランザクショントレースは，デフォルトではトランザクションブランチがロールバックおよびヒューリスティック決着した場合に記録します。トレースに記録する条件は，トランザクショントレース定義で変更できます。

3.2 回復処理

システムの一部に障害が発生しても，TPBroker では部分回復処理をしてシステムが全面停止しないようにしています。また，重大な障害が原因で全面停止した場合でも，全面回復処理をして，システムの状態やユーザの資源を回復できます。

3.2.1 部分回復処理

アプリケーションプログラムに障害が発生した場合，TPBroker はアプリケーションプログラム単位の回復処理（部分回復処理）をして，システム全体に影響がないようにします。

アプリケーションプログラムに障害が発生した場合，トランザクション内のアプリケーションプログラムプロセスが異常終了したことを TPBroker が検知します。そして，該当するトランザクションを回復して決着させます。

回復処理をしようとしているときにシステムが使用する領域が不足してトランザクションを回復できない場合があります。これを防ぐために，TPBroker は一定間隔で回復していないトランザクションを検索して，再び回復する処理をします。

3.2.2 全面回復処理

TPBroker のシステムに障害が発生した場合，前回のオンラインの状態を引き継いで，履歴情報を基に TPBroker システムを障害発生直前の状態に回復できます。この開始モードを再開始モードといいます。

(1) 全面回復処理の手順

TPBroker は再開始時に全面回復処理を実行します。全面回復処理はシステム回復処理，トランザクション回復処理の順で実行されます。

(a) システム回復処理

環境変数 TPFS に設定されたディレクトリ下に作成されたステータスファイルにはシステム制御情報が格納されています。まず，システム制御情報を基に，トランザクションの同期点処理と関係のないシステムの状態を回復します。この時点で，TPBroker はトランザクションサービスを再開します。

(b) トランザクション回復処理

新たなトランザクションサービス処理と並行して，システム障害で TPBroker が停止したときに実行中だったアプリケーションプログラムのトランザクション処理を回復して，ロールバックまたはコミットします。トランザクションをロールバックするのかコミットするのかは，トランザクション処理がどこまで進んでいたかで決まります。トランザクション処理が，2 相コミットのうちの 1 相目の完了前までの場合には，グローバルト

ランザクションをロールバックします。2 相コミットのうちの 1 相目が完了していた場合には，ルートトランザクションブランチでの決定に従ってグローバルトランザクションをロールバック，またはコミットします。

3.2.3 決着コマンドによる回復処理

全面回復処理，または部分回復処理をするとき，回復対象のトランザクションが prepared 状態の場合，TPBroker はコミットとロールバックのどちらに決着すればよいかが判断できません。このようなトランザクションは未決着のまま残りますので，どちらに決着すればよいかを決着コマンドを使って TPBroker に指示してください。コミットに決着する場合は tscommit コマンドを，ロールバックに決着する場合は tsrollback コマンドを入力してください。

また，トランザクションの状態は tslstrn コマンドを使用して参照できます。

これらのコマンドの詳細については，「9.3　運用コマンドの詳細」を参照してください。

3.2.4 障害のケース

TPBroker の障害には次のような原因があります。

(1) OTS が稼働していない場合

OTS が稼働していない場合，すべての OTS のインタフェースやコマンドはそのことを検知し，エラーリターンするか，または例外を発生させます。

C++ OTS の場合，クライアントがトランザクショナルサーバを呼び出し，サーバ側の OTS が稼働していない場合，トランザクショナルコールは失敗します。この場合，呼び出しがトランザクショナルでないときは，成功します。

Java OTS の場合，OTS が稼働していないとトランザクションコンテキストサーバはダウンし，トランザクションの処理はできません。

(2) アプリケーションプログラム異常終了

アプリケーションプログラムが異常終了すると，OTS はそのことを検知し，自動的にトランザクションを回復します。

(3) OTS デーモン異常終了

OTS デーモンが異常終了したあとに OTS デーモンを再開始すると，トランザクション処理を行うアプリケーションプログラムは OTS によって強制終了されます。再開始した OTS はトランザクションを回復します。ただし，一度もトランザクション処理をしていないアプリケーションプログラムは，異常終了しません。

(4) 決着デーモンおよび回復デーモン異常終了

決着デーモンおよび回復デーモンが異常終了すると，OTS デーモンはそのことを検知し，デーモンを再生成します。新しいデーモンは異常終了したデーモンからトランザクション処理を引き継ぎます。

決着デーモンおよび回復デーモンが，それぞれ 3 分間に 45 回を超えて連続異常終了した場合は，OTS デーモンが停止します。

(5) ORB 異常終了（C++）

OTS が ORB システムの異常終了を検知すると，OTS はすべての決着デーモン，回復デーモン，およびトランザクションを実行しているアプリケーションプログラムを強制終了し，OTS 自身も異常終了します。

なお，この処理は，プロセス監視定義がデフォルト値の場合の処理です。また，この処理は，C++ 実行環境でだけ行われます。

(6) マシンダウン

マシンダウンのあとで OTS が起動されると，OTS は自動的にトランザクションを回復します。

(7) ヒューリスティック決着

OTS の運用コマンドで，トランザクションの状態を決着させてください。

4 C++ OTS 機能（C++）

TPBroker の OTS には，C++ 実行環境および Java 実行環境で使用できる OTS と，C++ 実行環境だけで使用できる C++ OTS と，Java 実行環境だけで使用できる Java OTS があります。この章では，C++ 実行環境だけで使用できる C++ OTS の機能について説明します。
C++ 実行環境および Java 実行環境で使用できる OTS については，「3.　OTS 機能」を参照してください。
Java 実行環境だけで使用できる Java OTS については，「5.　Java OTS 機能（Java）」を参照してください。
C++ OTS 機能は，Cosminexus TPBroker ではサポートしていません。

4.1　トランザクションマネージャ機能

TPBroker はリソースマネージャと接続してトランザクションを処理できます。

4.1.1　トランザクション処理との関係

トランザクション処理とリソースマネージャを連携させるには，X/Open で規定した DTP モデルの XA インタフェースをサポートするようにします。

リソースマネージャを使う場合，そのリソースマネージャが XA インタフェースをサポートしていれば，TPBroker のトランザクションと連携できます。サポートしていないと，TPBroker のトランザクションと同期を取ることができません。

リソースマネージャの運用については，「7.4　リソースマネージャの運用（C++）」を参照してください。

4.1.2　XA インタフェース

TPBroker は XA インタフェースのすべての関数をサポートしていますが，マイグレーション機能についてはサポートしていません。

(1)　リソースマネージャとの連携

TPBroker と XA インタフェースを使用して，DBMS などのリソースマネージャと連携し，トランザクションを処理できます。リソースマネージャと連携する場合は，「7.4.1 XA インタフェースをサポートしたリソースマネージャの場合」に示す手順に従ってください。

(2)　マルチスレッド対応

TPBroker の XA インタフェースサポートはマルチスレッドに対応しているため，マルチスレッド環境で XA インタフェースによってリソースマネージャと連携できます。TPBroker はシングルスレッド環境をサポートしていません。リソースマネージャによっては XA ライブラリがマルチスレッドに対応していないものがあり，この場合，TPBroker と連携できません。

マルチスレッド環境で XA インタフェースによってリソースマネージャと連携する場合，リソースマネージャによって，次のような違いがあります。

- プロセスごとに xa_open ／ xa_close 関数を発行する。
- スレッドごとに xa_open ／ xa_close 関数を発行する。

TPBroker では，リソースマネージャ定義 set_xa_open_scope（xa_open 発行単位）および set_xa_open_timing（xa_open 発行タイミング）を設定することで xa_open および xa_close 関数の発行のタイミングを変更し，リソースマネージャと XA インタフェース

で連携できるようにしています。

(3) OTS とのモデルインタオペラビリティ

OTS インタフェースで作成されたアプリケーションプログラムを，XA インタフェースを使用してリソースマネージャと連携させることができます。ただし，OTS のアプリケーションモデルは，フラットトランザクション，間接コンテキスト管理，および暗黙的プロパゲーションを使用したものに限ります。

したがって，XA インタフェースを使用してリソースマネージャと連携する場合は，ネスティッドトランザクションは使用できません。また，直接コンテキスト管理および明示的プロパゲーションを使用する場合は，トランザクションをスレッドに関連づけてから（Current オブジェクトの resume オペレーションを発行してから）リソースマネージャにアクセスする必要があります。

4.1.3 TX インタフェース

TPBroker は，TX インタフェースのすべての関数をサポートしています。ただし，次に示す項目はサポートしません。

- commit_return 属性のうち，TX_COMMIT_DECISION_LOGGED 特性

(1) OTS とのモデルインタオペラビリティ

TX インタフェースを使用して開発したアプリケーションプログラムは，OTS でフラットトランザクション，間接コンテキスト管理，および暗黙的プロパゲーションを使用したアプリケーションモデルと同等です。このため，同等なアプリケーションモデルで開発された OTS アプリケーションとインタオペラビリティがあります。例えば，TX インタフェースを使用して開発したクライアントアプリケーションが tx_begin 関数の呼び出し後，ADAPTS または REQUIRES の OTSPolicy を設定したサーバに対してリクエストを発行すると，暗黙的プロパゲーションによってトランザクションコンテキストがプロパゲートされます。また逆に，OTS インタフェースで開発されたクライアントアプリケーションがトランザクションを開始したあと，ADAPTS または REQUIRES の OTSPolicy を設定したサーバに対してリクエストを発行して暗黙的プロパゲーションが行われると，tx_info 関数によってトランザクション情報を取得できます。ただし，TX インタフェースはネスティッドトランザクションをサポートしていないため，ここで TX インタフェースでトランザクションの開始（tx_begin 関数）を行うことはできません。

4.2　時間監視機能

TPBroker は，OTS の時間監視機能として次に示す機能を提供します。なお，リクエストの応答時間監視機能については，ORB が提供するため，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

4.2.1　トランザクション処理時間監視

TPBroker は，アプリケーションプログラムが，あるトランザクションを開始してから，そのトランザクションに対する処理を完了するまでの時間を監視します。

あらかじめそのトランザクションに対して設定された監視時間を経過した場合，そのトランザクションを実行中のアプリケーションプログラムのプロセスを強制停止させます。プロセスが強制停止されると，プロセス監視機能によってそのトランザクションをロールバックします。プロセスを強制停止するため，監視時間を経過したトランザクション以外にも，その時点で同じプロセス上で処理中であったすべてのトランザクションがロールバックされます。ここで処理の完了とは，トランザクションを開始したアプリケーションプログラムが，トランザクションの完了指示（コミット指示またはロールバック指示）を発行することをいいます。この監視時間は，暗黙的プロパゲーションによってトランザクションコンテキストとともにサーバアプリケーションにプロパゲーションされます。この場合のサーバアプリケーションでのトランザクション処理の開始は，トランザクショナルリクエストを受信した時点のことをいい，そのリクエストに対する応答を返す時点のことをトランザクション処理の完了といいます。この場合，サーバアプリケーションでも同様の時間監視が行われます。

また，トランザクションコンテキストの管理方式に関係なく監視されるため，そのスレッドからサスペンド（Current::suspend()）されているトランザクションも対象となります。トランザクション処理時間監視の監視時間の設定は，トランザクション開始時に Current，TransactionFactory，TX インタフェースで設定できます。設定の方法については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。また，どの方法によっても監視時間の設定をしない場合は，トランザクション処理時間の監視はしません。

4.2.2　トランザクション決着指示待ち時間監視

トランザクション処理時間監視機能では，クライアントアプリケーションがコミット要求またはロールバック要求を行うまでの時間，またはサーバアプリケーションがリクエストを受信してから応答を返すまでの時間を監視します。ただし，サーバアプリケーションで処理されたトランザクションに対する完了指示は監視対象外であるため，通信障害やクライアントノードのシステム障害などによって，そのトランザクションに対する決着指示を長時間受信できない場合があります。このような場合，そのトランザクション処理でアップデートされたリソースを長時間占有してしまい，システム全体のスループットの低下を招くことになります。このため，TPBroker ではサーバアプリケー

ションがそのトランザクションに対する処理を完了してから，2 相コミットの 1 相目の指示（通常はプリペア指示）を受信するまでの時間を監視し，一定時間を経過したものについてはそれ以上決着指示を待たないで自動的にロールバックします。この監視は常に行われ，監視時間は 180 秒です。

また，プリペア指示の受信後，2 相目の指示（コミット指示またはロールバック指示）の受信を待つトランザクションについては，時間監視によるロールバックはしません。

4.2.3 トランザクションサスペンド時間監視

明示的プロパゲーションによってトランザクションをプロパゲーションする場合は，サーバアプリケーションがリジュームしたトランザクションをサスペンドした状態のまま，クライアントに応答を返すことができます。このタイミングでクライアントノードのシステム障害などが発生すると，トランザクションが長時間サスペンドされたままとなります。このため，アプリケーションプログラムがトランザクションをサスペンド（Current::suspend()）してからリジューム（Current::resume()）するまでの間は，トランザクション処理時間監視とは関係なく，TPBroker が時間監視を行います。

この監視は常に行われ，監視時間は 180 秒です。監視時間を経過した場合，そのトランザクションをサスペンドしたアプリケーションプログラムのプロセスを強制停止させることで，トランザクションをロールバックさせます。アプリケーションプロセスを強制停止させるのは，サスペンドされているトランザクションの処理でアプリケーションプログラム内に確保された OTS やリソースマネージャの資源を解放するためです。

また，Current::begin() によってサブトランザクションが生成された場合，それ以前にそのスレッドに関連していたトランザクションは自動的にサスペンド状態となります。このため，このトランザクションも監視の対象となります。ネスティッドトランザクションを使用する場合は注意が必要です。

4.3　API トレースの取得

ここでは，API トレースの取得や自動削除などについて説明します。

4.3.1　概要

API トレースとは，OTS が提供する API の出入口で取得される，API 実行の履歴のことです。OTS が実装するオブジェクトの持つ主な API の履歴を取得します。OTS は取得した API トレースを API トレースファイルに格納します。API トレースファイルは運用コマンドで解析し，標準出力に出力できます。

4.3.2　トレースの取得

API トレースを取得するには，環境変数 TPSYS_APTUSE を設定します。取得された API トレースは，スレッドごとに API トレースファイルに格納されます。

API トレースファイルはスレッド 32 個分の容量を持ち，プロセス一つにつき最低 1 個，最大 128 個生成され，ファイル名に通番を付けて管理されます。まず，最初の API 発行やトランザクショナルサーバの呼び出しなどで，API トレースファイルが一つ生成されます。このあと，プロセス中で API 発行を同時に行うスレッド数が 32 個増えるごとに，新しく API トレースファイルが生成されます。

API トレースファイルのフルパスは次のようになります。

```
$TPSPOOL/aptrace/aptプロセスID_プロセス作成時分秒_プロセス名_通番
```

プロセス作成の時・分・秒は 2 けたにゼロパディングされます。

例　`otsspool/aptrace/apt190_030502_orig_0`

また，スレッドごとに環境変数 TPSYS_APTTRCPERTHRD で設定した個数の API トレースを格納できます。一つのスレッドからこれ以上の API が発行された場合には，古い API トレースから上書きされます。

API トレースは API の入り口と出口で必ず取得されますが，トレース容量の節約のため，入り口トレースは出口トレースによって上書きされます。

4.3.3　トレースの解析

API トレースファイルは tsedapt コマンドで解析されます。解析結果は出力して参照できます。コマンドの書式は「9.3　運用コマンドの詳細」を参照してください。

4.3.4　トレースファイルの自動削除

API トレースファイルは，次のようなタイミングで，システム環境定義に従って削除されます。

- TPBroker 開始時（正常開始および再開始）
- TPBroker 開始後，一定時間（トランザクション定義 /OTS/set_trace_remove_interval で指定した秒数）ごと

API トレースファイルの削除の規則を次に示します。

- トランザクション定義 /OTS/max_trace_remain_num で指定した数を超える API トレースファイルがない場合は削除しません。
- API トレースファイルの最終更新時間が古いものから優先的に削除します。
- 動作中のプロセス（TPBroker とは関係のないプロセスを含む）と同一のプロセス ID（pid）をファイル名に持つ API トレースファイルは，削除できないとみなされます。
- API トレースファイルの残存個数がトランザクション定義 /OTS/max_trace_remain_num で指定した個数になるように削除します。ただし，削除できるファイルをすべて削除した場合は，この個数と一致しないときでも削除を終了します。

例

動作中プロセス：Explorer:pid=1，server1:pid=2，client:pid=3
残存ファイル：apt1，apt2，apt3，apt4，apt5，apt6（最終更新時間の古い順）

- トランザクション定義：/OTS/max_trace_remain_num=4 の場合
  TPBroker の開始からトランザクション定義 /OTS/set_trace_remove_interval で指定した秒数が経過すると，apt1，apt2 および apt3 は動作中プロセスと同一のプロセス ID を持つので削除できないため，apt4 および apt5 が削除されて apt1，apt2，apt3 および apt6 が残ります。
- トランザクション定義：/OTS/max_trace_remain_num=2 の場合
  TPBroker の開始からトランザクション定義 /OTS/set_trace_remove_interval で指定した秒数が経過すると，apt1，apt2 および apt3 は動作中プロセスと同一のプロセス ID を持つので削除できないため，/OTS/max_trace_remain_num=2 の指定に反して apt1，apt2 および apt3 の三つが残ります。

4.3.5　使用上の注意

API トレースを取得するときは，次の点に注意してください。

API トレースファイルの最大個数

API トレースファイルはプロセス当たり最大 128 個生成されます。この個数を超えた場合，イベントログまたは syslog にメッセージ KFCB30401-W が出力されます。このメッセージが出力されたプロセスの API トレースは取得されません。

API トレースファイル解析コマンドの発行タイミング

取得した API トレースは原則的にアプリケーションプログラムの停止後に解析するものとし，アプリケーションプログラム実行中の解析結果は保証されません。ただし，アプリケーションプログラム実行中に解析しても，アプリケーションプログラムに影響はありません。

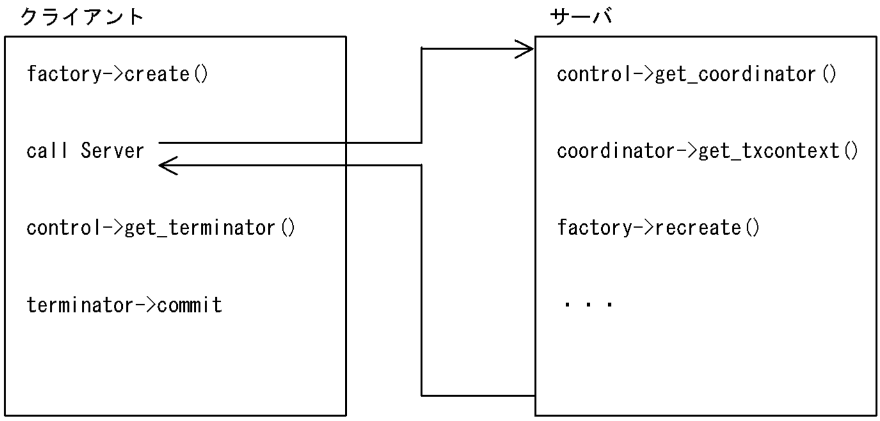
API トレース取得対象オブジェクト

削除されたオブジェクトに対する API コールには例外が発生しますが，この例外に関する API トレースは取得できません。また，Resource，SubtransactionAwareResource，Synchronization，および RecoveryCoordinator オブジェクトでは API トレースは取得されません。

API トレースが取得されるプロセス

API トレースは，呼び出されたオブジェクトを作成したプロセスで取得されます。例えば，次の図に示すような構成の明示的プロパゲーション時のインターポジションでは，サーバによる get_coordinator オペレーションの呼び出しは，クライアントで作成された Control オブジェクトに対して行われるため，API トレースはクライアントプロセスで取得されます。同様に，get_txcontext オペレーションの呼び出しもクライアントプロセスで取得されます。

図 4-1　サーバでの API コールがクライアントで取得される例



4.4　高速オプションライブラリ（OTS Fast Path Option）

ここでは，TPBroker の OTS 機能のオプションである高速オプションライブラリ（OTS Fast Path Option）について説明します。

4.4.1　OTS Fast Path Option とは

OTS Fast Path Option は，OTS の基本的な機能だけを使用するシンプルなアプリケーションプログラムを実行するための OTS ライブラリです。機能に制限がある代わりに，軽量で高速な OTS 機能を提供します。

通常の OTS では，ネスティッドトランザクションや直接コンテキスト管理など，自由度の高いトランザクションのコンテキスト管理をサポートしています。このため，さまざまなモデルのアプリケーションプログラムを開発できます。また，ほかの OTS 製品とのインタオペラビリティがあります。しかし，多くのシステムで使用されているシンプルなアプリケーションプログラムにとっては，不要な処理が含まれることになります。

OTS Fast Path Option は，このような機能を制限して処理の最適化を図り，高速に処理できるようにしています。

4.4.2　OTS Fast Path Option の特長

OTS Fast Path Option の特長を次に示します。

- 通常の OTS と同様に，XA インタフェースで DBMS のようなリソースマネージャと連携してトランザクションを処理できます。
- API は OMG の OTS 1.3 と完全に互換します。ただし，使用できる API の種類には制限があります。
- 通常の OTS とはライブラリの切り替えで区別するため，同一システムおよび同一ノード内で共存できます。ただし，相互のアプリケーションプログラムの連携はできません。
- マルチスレッドライブラリを提供しています。
- アプリケーションプログラム開発手順やシステム運用方法などは，通常の OTS ライブラリを使用するときと同じです。ただし，ライブラリ名は異なります。

4.4.3　OTS Fast Path Option の制限事項

OTS Fast Path Option を使用するときに制限される機能および API について説明します。

(1)　制限される機能

OTS Fast Path Option で制限される機能を次の表に示します。

表 4-1　OTS Fast Path Option で制限される機能

機能	OTS Fast Path Option	通常の OTS
TX インタフェース	なし	あり
Resource オブジェクト（SubtransactionAwareResource を含む）	登録および使用できない	登録および使用できる
Synchronization オブジェクト	登録および使用できない	登録および使用できる
ネスティッドトランザクション	使用できない	使用できる
コンテキスト管理方式	間接方式だけ	間接方式または直接方式
プロパゲーション方式	暗黙的方式だけ	暗黙的方式または明示的方式
リジュームチェック	なし※	あり
サスペンド時間監視	なし※	あり
OTS インタオペラビリティ	なし	あり

注※

直接コンテキスト管理をサポートしないため不要です。

表 4-1 に挙げた機能を使用しないアプリケーションプログラムの場合に，OTS Fast Path Option を使用できます。これ以外のアプリケーションプログラムでは，必ず通常の OTS を使用してください。

(2)　制限される API

機能の制限によって，API も制限を受けます。OTS Fast Path Option で制限される API について次に説明します。

OTS インタフェース

次のインタフェースに対してはオブジェクトリファレンスの取得方法が提供されないため，これらのインタフェースのすべてのオペレーションを使用できません。

- CosTransactions::TransactionFactory
- CosTransactions::Control
- CosTransactions::Coordinator
- CosTransactions::Terminator
- CosTransactions::RecoveryCoordinator

CosTransactions::Current インタフェースについては，次のオペレーションだけが使用できません。次のオペレーションを実行した場合は，CORBA::IMP_LIMIT 例外が発生します。

- CosTransactions::Current::get_control()
- CosTransactions::Current::suspend()
- CosTransactions::Current::resume()

ORB インタフェース

特に制限はありません。

TPBroker 拡張インタフェース

TpCosOTS インタフェースについては，次のオペレーションだけが使用できません。次のオペレーションを実行した場合は，CORBA::IMP_LIMIT 例外が発生します。

- TpCosOTS::get_factory()

TX インタフェース

TX インタフェースはサポートされません。すべての TX 関数の呼び出しに対して，戻り値として TX_FAIL が返されます。

4.4.4　アプリケーションプログラムの開発手順

アプリケーションプログラムの開発手順は，通常の OTS ライブラリを使用するときと同じです。アプリケーションプログラムの開発手順およびインタフェースの詳細については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

4.4.5　使用上の注意

ここでは，通常の OTS で動作していたアプリケーションプログラムを OTS Fast Path Option で動作させる場合や，両者のプログラムを共存して使用する場合などの注意事項を説明します。

(1)　通常の OTS アプリケーションを OTS Fast Path Option で使用する

OTS Fast Path Option で制限された機能を使用していない場合に，通常の OTS で動作していたアプリケーションプログラムを OTS Fast Path Option で動作させることができます。この場合，OTS Fast Path Option ライブラリを指定して，アプリケーションプログラムをリンクし直してください。

なお，OTS Fast Path Option で制限されている機能を使用している場合でもリンクは成功しますが，制限された機能を実行しようとした時点で CORBA::IMP_LIMIT 例外が発生します（TX 関数の場合は，戻り値として TX_FAIL が返ります）。

通常の OTS から OTS Fast Path Option へのアプリケーションプログラムの移行は，使用機能の調査などを含めて慎重に行ってください。

(2)　通常の OTS アプリケーションとの共存

OTS Fast Path Option を使用したアプリケーションプログラムと，通常の OTS を使用

したアプリケーションプログラムを同時に使う場合は，次の制約がありますので注意してください。

- 複数のマシンで相互に通信する場合
  通常の OTS アプリケーションと OTS Fast Path Option を使用するアプリケーションプログラムとの間でトランザクションの連携（プロパゲーション）はできません。
  誤ってプロパゲーションを行った場合，サーバから CORBA::INVALID_TRANSACTION 例外が返され，トランザクションがロールバックされます。ただし，サーバ側の TPBroker が OTS Fast Path Option をサポートしていないバージョンの場合は，サーバアプリケーションが異常終了するときがあります。
- 同一マシン上の場合
  通常の OTS アプリケーションと OTS Fast Path Option を使用するアプリケーションプログラムとの間でトランザクションの連携（プロパゲーション）はできません。
  誤ってプロパゲーションを行った場合，サーバから CORBA::INVALID_TRANSACTION 例外が返され，そのトランザクションがロールバックされます。
- 同一プロセス（プログラム）上の場合
  通常の OTS ライブラリと OTS Fast Path Option を同時に使用（リンクまたはダイナミックロード）しないでください。同時に使用した場合は，動作は保証されません。

通常の OTS と OTS Fast Path Option との間のプロパゲーションを防ぐために，通常の OTS を使用したサーバアプリケーションと OTS Fast Path Option を使用したサーバアプリケーションとでは，必ず異なるオブジェクト名を使用してください。同じオブジェクト名を使用していると，クライアントアプリケーションからの _bind メソッドによって誤って接続してしまう可能性があります。

5 Java OTS 機能（Java）

TPBroker の OTS には，C++ 実行環境および Java 実行環境で使用できる OTS と，C++ 実行環境だけで使用できる C++ OTS と，Java 実行環境だけで使用できる Java OTS があります。この章では，Java 実行環境だけで使用できる Java OTS の機能について説明します。
C++ 実行環境および Java 実行環境で使用できる OTS については，「3.　OTS 機能」を参照してください。
C++ 実行環境だけで使用できる C++ OTS については，「4.　C++ OTS 機能（C++)」を参照してください。
Cosminexus TPBroker は, Java OTS を経由して Cosminexus Component Container と通信しますが，ユーザは意識する必要がありません。

5.1　Java OTS の構成

この節では，Java OTS のシステム構成について説明します。

5.1.1　基本構成

Java OTS 環境は次の三つから構成されています。

- ライトウェイト Java クライアント（Java アプリケーション）
トランザクションを開始，終了させますが，トランザクションコンテキストは保持しません。代わりに，クライアント内の ClientProxy オブジェクトが実際のトランザクションのローカルプロクシの役割を果たします。実際のトランザクションのトランザクションコンテキストは，トランザクションコンテキストサーバにあります。
- ライトウェイト Java サーバ（Java アプリケーション）
Java および C++ トランザクショナルクライアントからプロパゲーションされたトランザクションと結合します。しかし，トランザクションコンテキストは保持しません。代わりに，サーバ内の ServerProxy オブジェクトが実際のトランザクションのローカルプロクシの役割を果たします。実際のトランザクションのトランザクションコンテキストは，トランザクションコンテキストサーバにあります。
- トランザクションコンテキストサーバ（TCS）
ライトウェイト Java トランザクショナルクライアント，およびサーバのトランザクションコンテキストを保持します。

ライトウェイト Java トランザクショナルクライアントがトランザクションを開始する場合，クライアントマシンはトランザクションコンテキストを保持しません。その代わりに，Java OTS が ClientProxy オブジェクトを作成します。このオブジェクトは実際のトランザクションに対してローカルプロクシとしての役割を果たします。ClientProxy オブジェクトは，すべてのトランザクション管理をトランザクションコンテキストサーバに任せます。ClientProxy オブジェクトはトランザクション情報をキャッシュするので，性能が向上します。

ライトウェイト Java トランザクショナルサーバがトランザクション要求を受け取ると，サーバはトランザクショナルオブジェクトに対して ServerProxy オブジェクトを作成することによって，プロパゲーションされたトランザクション中にサーバ自身を介在させます。

ClientProxy および ServerProxy オブジェクトは，実際のトランザクションに対してローカルプロクシとしての役割を果たします。実際のトランザクションのトランザクションコンテキストは，トランザクションコンテキストサーバに保持されます。

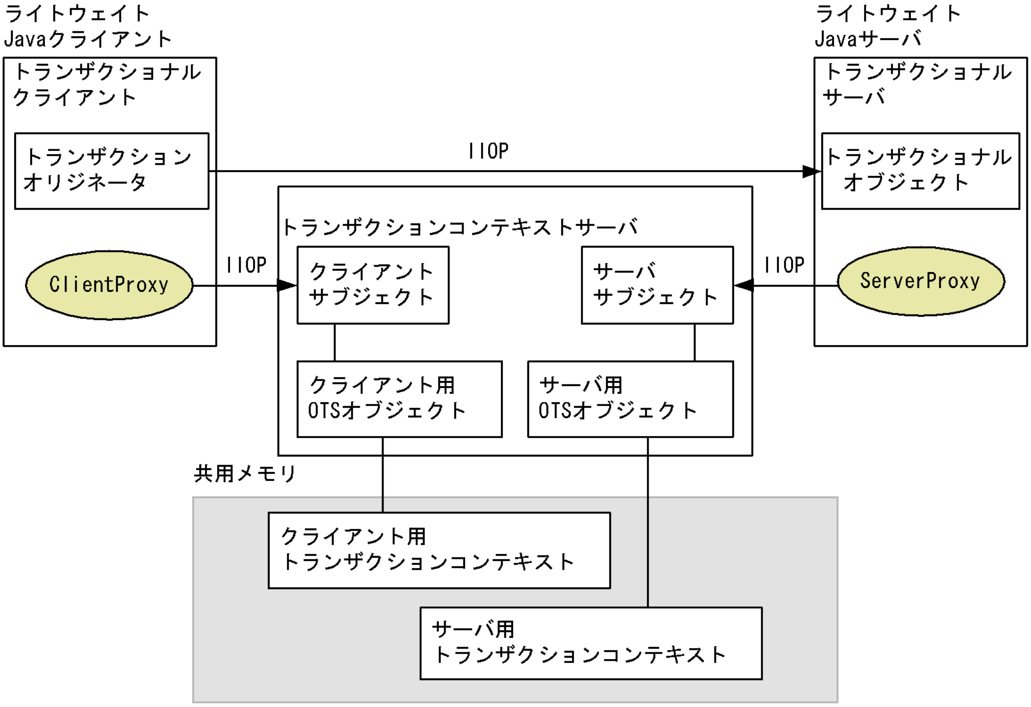
トランザクションコンテキストサーバは，ClientProxy および ServerProxy オブジェクトからの要求に従って，トランザクションコンテキストを保持および管理します。トランザクションコンテキストサーバは，トランザクションコンテキストを管理するために，

OTS 環境を用意する必要があります。トランザクションコンテキストサーバと OTS 環境は同じマシン上になければなりません。

ClientProxy および ServerProxy オブジェクトは，IIOP プロトコル方式によってトランザクションコンテキストサーバと通信するため，クライアント，サーバ，およびトランザクションコンテキストサーバを三つの個別のマシン上に実装できます。三つすべてを一つのマシン上に実装することもできますが，より良い性能を得るには，トランザクショナルクライアントをクライアントマシンに，トランザクショナルサーバとトランザクションコンテキストサーバをサーバマシンに実装するのが一般的です。

Java OTS 構成の概要について次の図に示します。

図 5-1　Java OTS 構成の概要

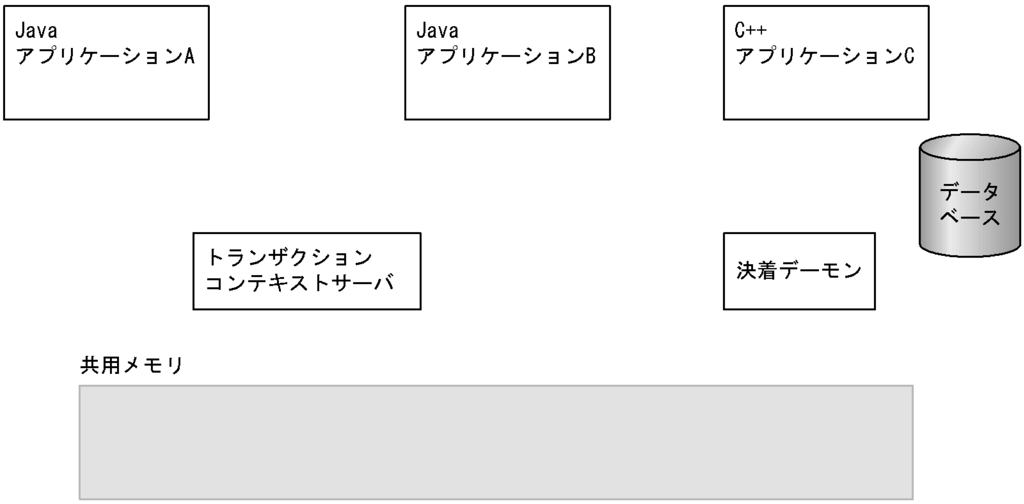


5.1.2　Java OTS について

この項では，Java OTS が分散トランザクションを管理する方法を説明します。

一般的な Java のインプリメンテーションの例を次の図に示します。この例には三つの分散アプリケーションがあり，そのうち二つは Java で，一つは C++ です。C++ アプリケーションはデータベース（DB）に接続しています。トランザクションコンテキストサーバおよび決着デーモンは OTS 環境が提供します。

図 5-2　一般的な Java のインプリメンテーション



関連オブジェクトはすべて，TPBroker によってその ACID 特性を保証されます。

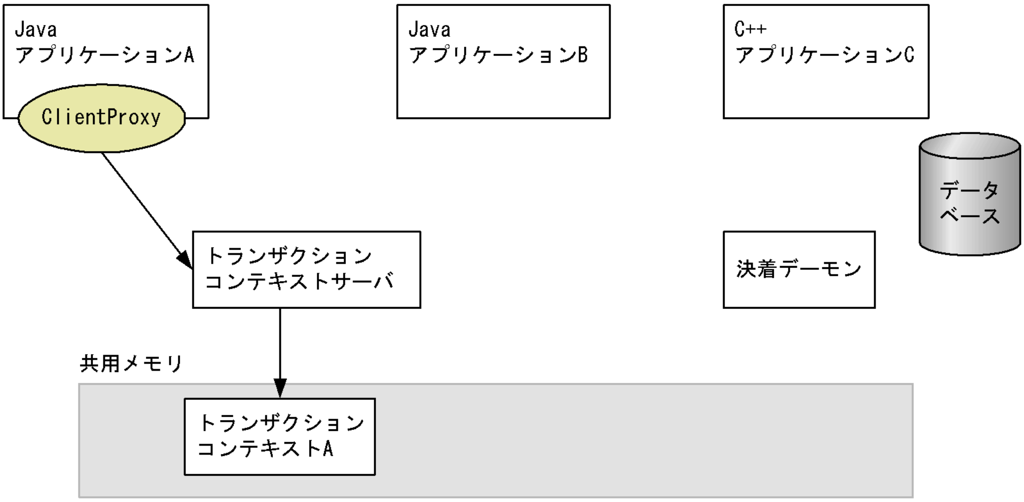
図 5-2 には一つの共用メモリ領域が示されていますが，各アプリケーションプログラムが異なるマシンにある場合は，共用メモリも別々となります。アプリケーションプログラムが共用メモリ領域を共用する必要はありません。

図 5-2 の例では，Java アプリケーション A は Java アプリケーション B を呼び出し，次に C++ アプリケーション C を呼び出します。そして，C++ アプリケーション C はデータベースを更新します。すべてのオペレーションは間接コンテキスト管理（Current インタフェースを使用）と暗黙的プロパゲーションを使用します。この処理の流れの詳細を，次に示します。

(1) トランザクションの開始

トランザクションの開始を次の図に示します。

図 5-3　トランザクションの開始

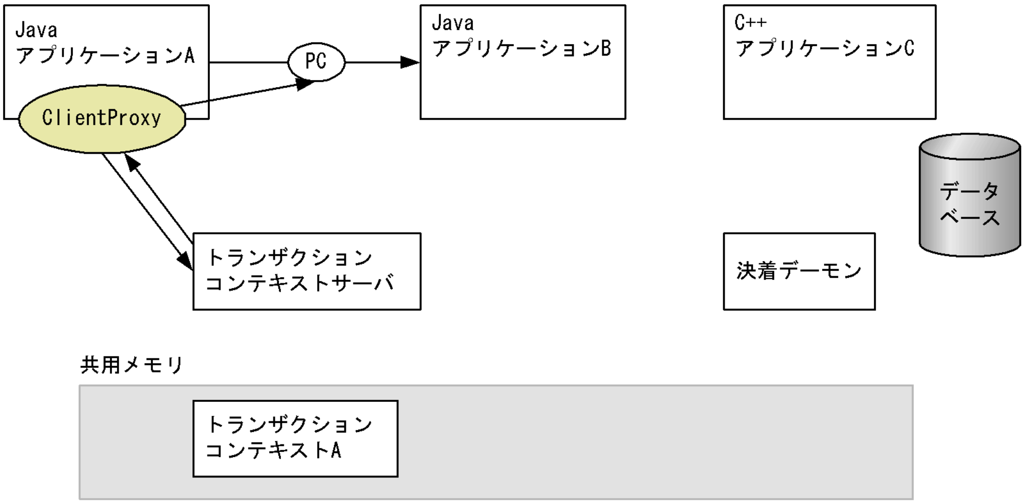


Java アプリケーション A はトランザクションを開始します。Java OTS は ClientProxy オブジェクトを作成します。これには，新しいトランザクションを作成するトランザクションコンテキストサーバが必要です。トランザクションコンテキストサーバは，共用メモリ内に新しいトランザクションコンテキストを作成します。

(2)　Java アプリケーションへのトランザクションのプロパゲーション

Java アプリケーションの呼び出しとトランザクションのプロパゲーションを次の図に示します。

図 5-4　Java アプリケーションの呼び出しとトランザクションのプロパゲーション



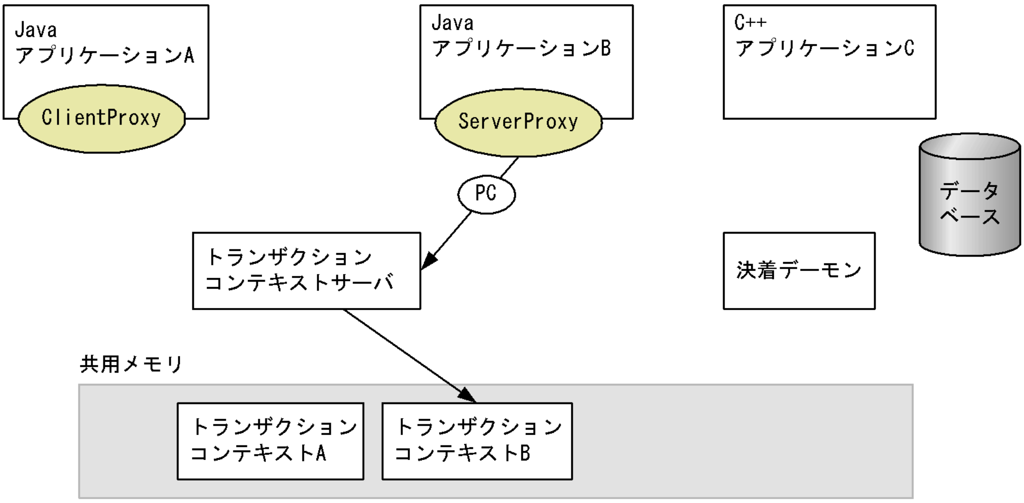
（凡例）
PC：プロパゲーションコンテキスト

Java アプリケーション A は Java アプリケーション B へトランザクションをプロパゲーションします。ClientProxy オブジェクトはオブジェクト呼び出しを検出し，トランザクションコンテキストサーバからプロパゲーションコンテキストを取得し，それをトランザクションリクエストに付加します。

(3)　ブランチトランザクションの開始

ブランチトランザクションの開始を次の図に示します。

図 5-5　ブランチトランザクションの開始



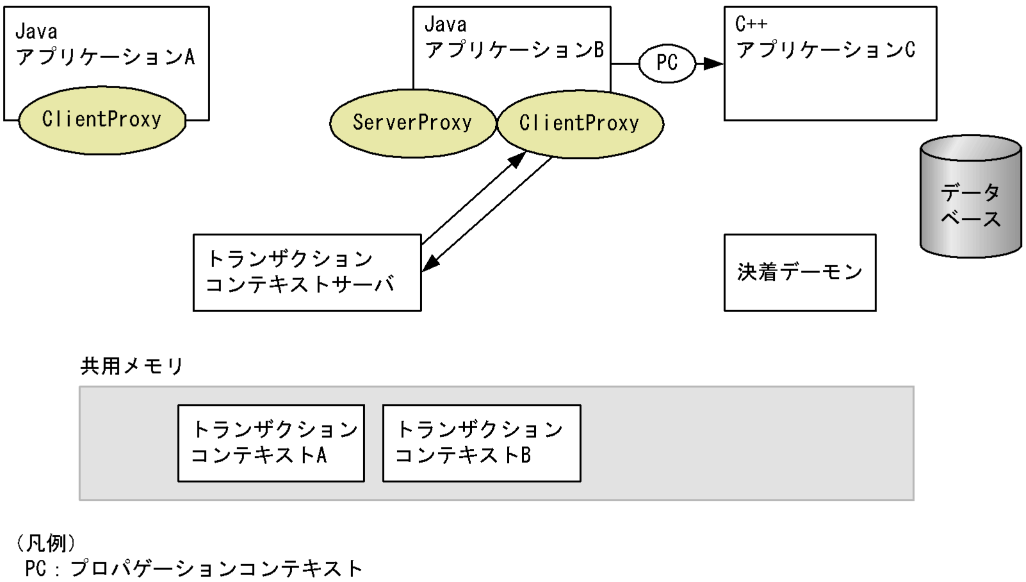
（凡例）
PC：プロパゲーションコンテキスト

Java アプリケーション B は，プロパゲーションコンテキストを受け取り，図 5-4 で示したプロパゲーションコンテキストと同じプロパゲーションコンテキストをトランザクションコンテキストサーバに渡します。そして，トランザクションコンテキストサーバは Java アプリケーション B 用にブランチトランザクションコンテキスト（トランザクションコンテキスト B）を作成します。

(4)　C++ アプリケーションへのトランザクションのプロパゲーション

C++ アプリケーションへのトランザクションのプロパゲーションを次の図に示します。

図 5-6　C++ アプリケーションへのトランザクションのプロパゲーション

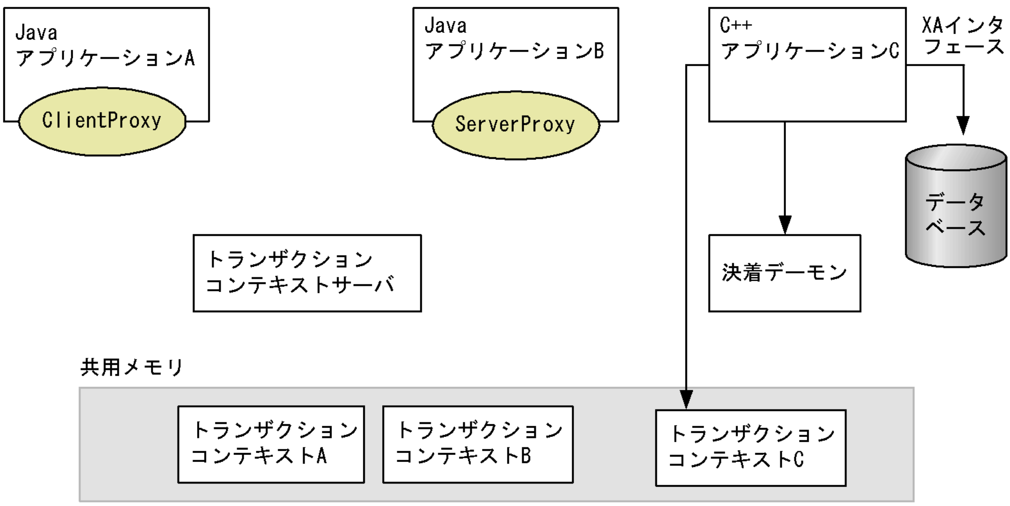


Java アプリケーション B は，プロパゲーションコンテキストを C++ アプリケーション C に渡すことによって，C++ アプリケーション C を呼び出します。ServerProxy オブジェクトはオブジェクト呼び出しを検出し，トランザクションコンテキストサーバからプロパゲーションコンテキストを取得し，それを C++ アプリケーション C に渡します。

(5)　データベースの更新

データベースの更新を次の図に示します。

図 5-7　データベースの更新

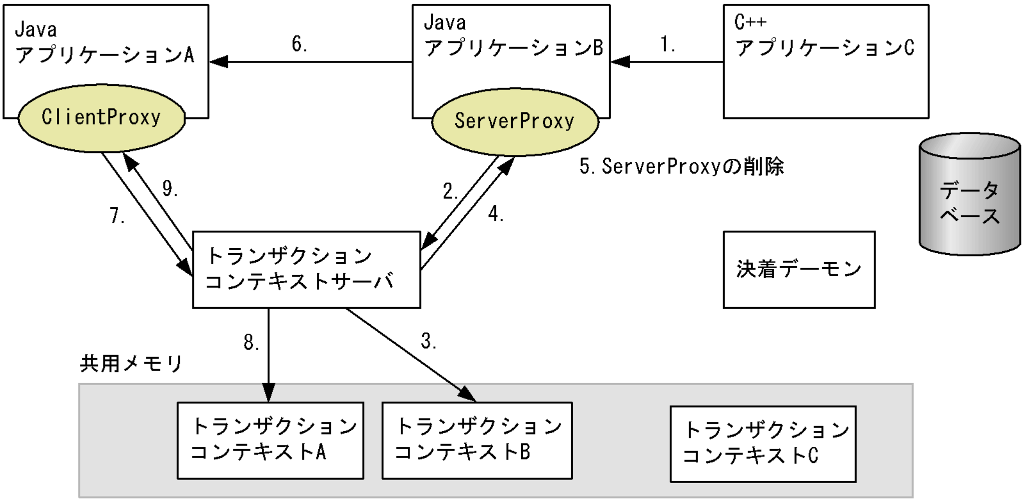


C++ アプリケーション C は，プロパゲーションコンテキストを受け取り，自動的にブランチトランザクションコンテキスト（トランザクションコンテキスト C）を作成します。そして，C++ アプリケーション C は XA インタフェースを介して，そのデータベースを更新します。

(6) アプリケーションプログラムへのオペレーションの返送

アプリケーションプログラムへのオペレーションの返送を次の図に示します。図中の番号は通信の行われる順序です。

図 5-8　アプリケーションプログラムへのオペレーションの返送

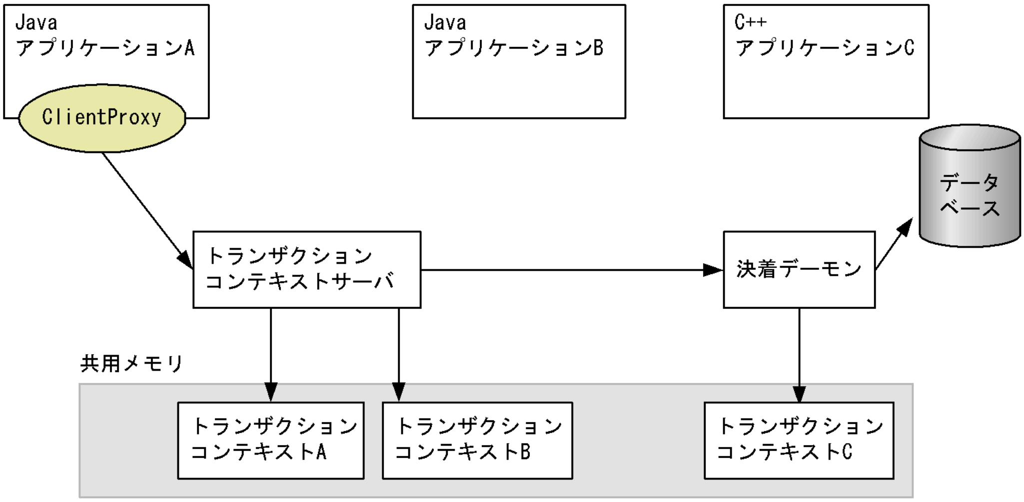


応答は Java アプリケーション A に戻されます。返送時に，Java OTS は Java アプリケーション B 上の ServerProxy オブジェクトを削除します。ServerProxy オブジェクトが削除されると，トランザクションコンテキストサーバは，トランザクションを決着させる責任を持ちます。Java アプリケーション A 上の ClientProxy オブジェクトは，Java アプリケーション B から応答を受けたことをトランザクションコンテキストサーバに知らせます。

(7) トランザクションのコミット

トランザクションのコミットを次の図に示します。

図 5-9　トランザクションのコミット



ClientProxy オブジェクトを使用して，Java アプリケーション A はトランザクションコンテキストサーバにトランザクションをコミットするよう伝えます。トランザクションコンテキストサーバは，2 相コミット手順でトランザクションを決着させます。

つまり，トランザクションコンテキストサーバが，データベースのコミットおよびトランザクションコンテキスト C の削除のために決着デーモンを起動します。また，トランザクションコンテキストサーバがトランザクションコンテキスト A および B を削除します。

5.2　Java OTS API の概要

Java OTS は，アプリケーションプログラムを作成するための API を提供しています。これらの API の概要を次に示します。

API の詳細については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

● CosTransactions モジュール
OTS 仕様で指定されている CosTransactions インタフェースを，CORBA 準拠のパッケージ名称で提供します。パッケージ名称は次のとおりです。

CORBA 準拠パッケージ名称
　org.omg.CosTransactions

● CosTransactions IDL ファイル
OTS 仕様で指定されている CosTransactions インタフェースを示す IDL ファイルを，CORBA 準拠のファイル名称で提供します。ファイル名称は次のとおりです。

CORBA 準拠のファイル名称を持つ CosTransactions IDL ファイル
　$TPDIR/idl/CosTransactions.idl

● TpCosOTS インタフェース
TpCosOTS インタフェースは TPBroker 独自の機能を提供します。これによって，Java アプリケーションは次のことができるようになります。

- TransactionFactory オブジェクトの取得
- Java OTS が提供する独自のメソッドへのアクセス

● org.omg.CORBA.ORB.resolve_initial_references メソッド
org.omg.CORBA.ORB.resolve_initial_references メソッドは，Current 擬似オブジェクトと TpCosOTS オブジェクトへのアクセスを提供します。このメソッドには次のパラメタがあります。

- TransactionalCurrent
  Current 擬似オブジェクト用のパラメタ
- CosOTS
  TpCosOTS オブジェクト用のパラメタ

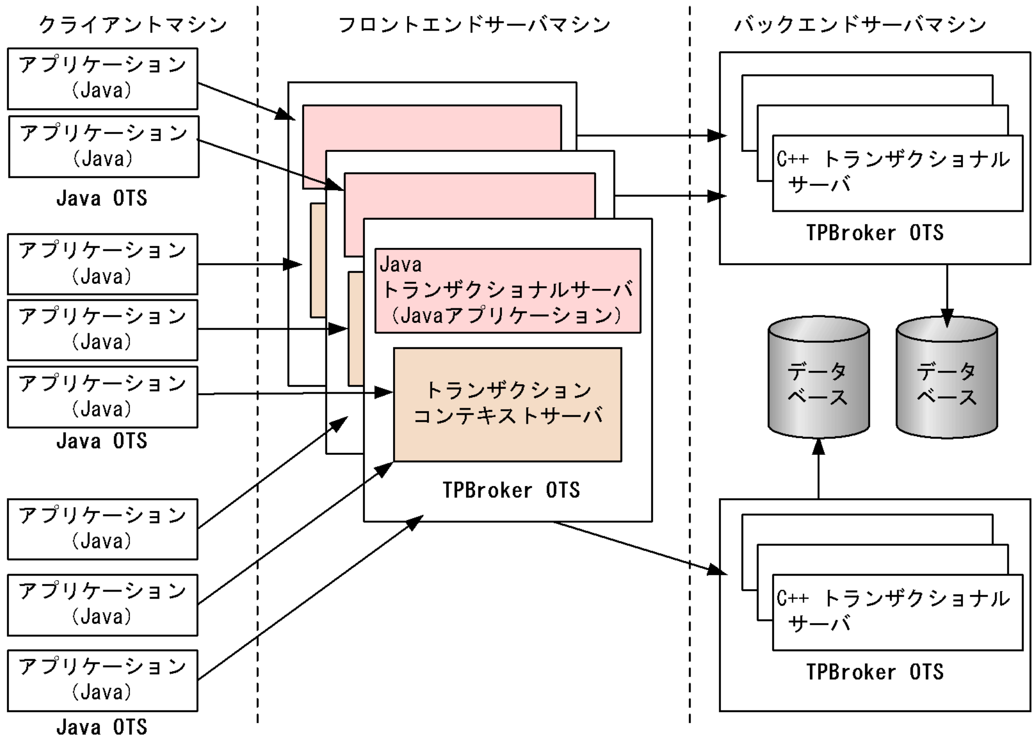
5.3　システム構成の選択

この節では，システムを設計する際の注意事項について説明します。

5.3.1　Java ベースのシステム構成

Java ベースのトランザクションの一般的なシステム構成を次の図に示します。Java トランザクショナルクライアントはクライアントマシンにあり，Java トランザクショナルサーバはフロントエンドサーバマシンにあります。より良い性能を得るために，それぞれのフロントエンドサーバマシンでは，Java トランザクショナルサーバとトランザクションコンテキストサーバが一緒に配置されることに注意してください。Java トランザクショナルサーバは C++ アプリケーションのバックエンドデータベースサーバにアクセスします。

図 5-10　Java ベースのトランザクションのシステム構成



5.3.2　トランザクションコンテキストサーバについて

ネットワークでは，最低一つのトランザクションコンテキストサーバが必要です。ネットワークとは ORB がアクセスできるドメインのことです。Java トランザクショナルクライアントおよび Java トランザクショナルサーバは，同じトランザクションコンテキス

トサーバを共用するか，または異なるトランザクションコンテキストサーバを使用するかのどちらかです。

5.3.3　Java アプリケーションについて

Java アプリケーションは，トランザクショナルクライアントまたはトランザクショナルサーバとしての役割を果たすことができます。Java アプリケーションとトランザクションコンテキストサーバを同じマシン上に配置する必要はありませんが，より良い性能を得るには，同じマシンに配置することをお勧めします。

5.4　トランザクションコンテキストサーバのネーミング

トランザクションコンテキストサーバは特定の名前と関連づけることができます。複数のトランザクションコンテキストサーバが同じ名前を持つこともできます。この名前はグループを指定し，そのグループとほかのグループとを区別します。

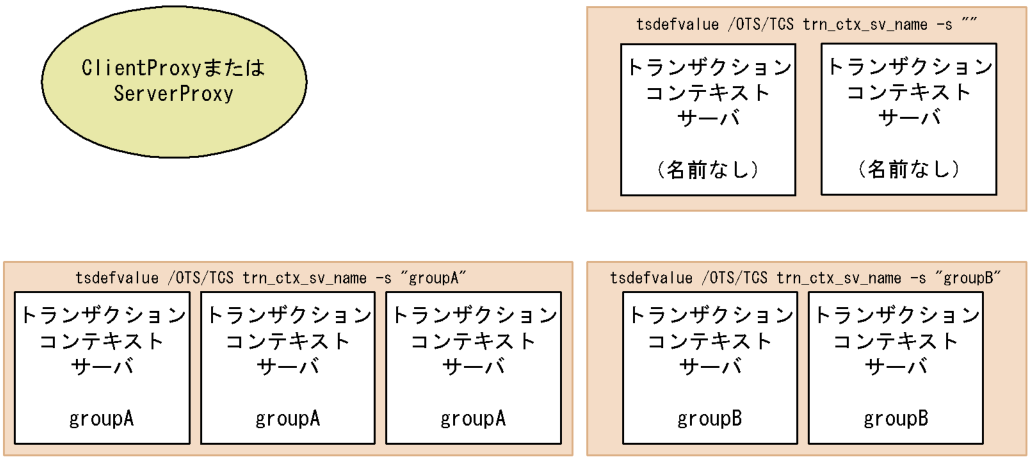
次のシステム環境定義で，トランザクションコンテキストサーバに名前を付けることができます。

```
/OTS/TCS trn_ctx_sv_name
```

Java トランザクショナルクライアントおよび Java トランザクショナルサーバでは，TpCosOTS オブジェクトの set_property メソッドを使用して，トランザクションコンテキストサーバの名前を設定できます。そして，ClientProxy または ServerProxy オブジェクトはこの名前を使用して，特定のトランザクションコンテキストサーバを見つけることができます。

トランザクションコンテキストサーバのネーミングについて次の図に示します。

図 5-11　トランザクションコンテキストサーバのネーミング



TCS 名が重要となるのは，次のような理由からです。

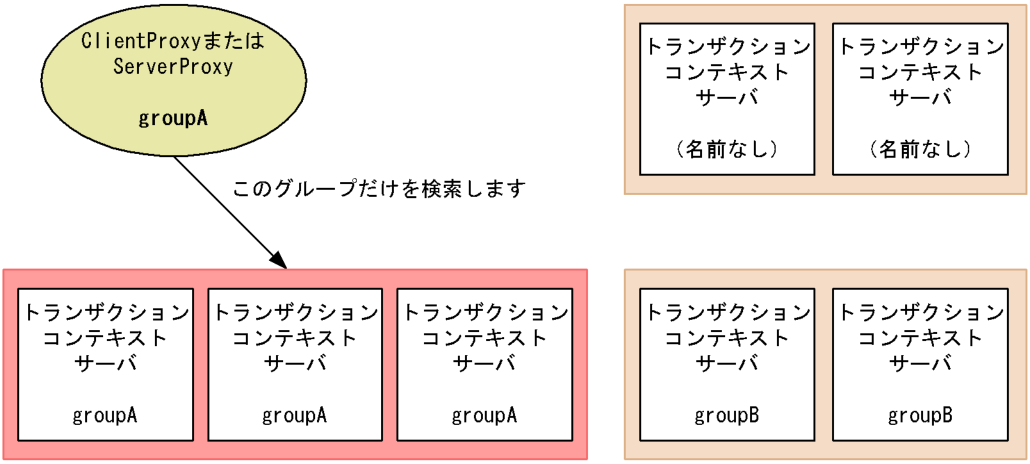
- 性能調整
- 効率的な負荷のバランス
- 開発環境とテスト環境の区別

ClientProxy または ServerProxy オブジェクトは，トランザクションコンテキストサーバの名前が設定されている場合，接続先のトランザクションコンテキストサーバがダウ

ンしていることを検出すると，同じグループ内にある別のトランザクションコンテキストサーバにバインドし直すことができます。これによって，複雑なシステムではより良い性能が得られます。

トランザクションコンテキストサーバグループを検索する例を次の図に示します。

図 5-12　トランザクションコンテキストサーバグループの検索（groupA を設定）

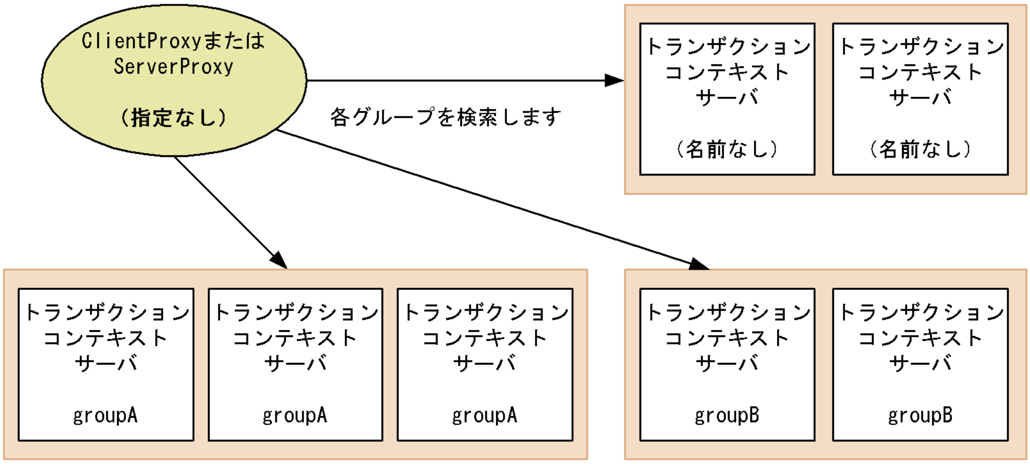


ClientProxy または ServerProxy オブジェクトは次のようにして名前を処理します。

- 名前が設定されている場合
  ClientProxy または ServerProxy オブジェクトは，その名前を持つすべてのトランザクションコンテキストサーバを検索し，その一つを選択してトランザクショナルオペレーションを実行します（図 5-12）。
- 名前が設定されていない場合
  ClientProxy または ServerProxy オブジェクトは，同じドメイン内にあるすべてのトランザクションコンテキストサーバを検索し，その一つを選択します（図 5-13）。

すべてのトランザクションコンテキストサーバを検索する例を次の図に示します。

図 5-13　すべてのトランザクションコンテキストサーバの検索（設定なし）



5.5 時間監視機能

Java OTS によって，トランザクション処理時間の監視ができます。

関連するトランザクションがタイムアウトすると，OTS 環境はトランザクションをロールバックします。TPBroker では，タイムアウト値はトランザクションコンテキストサーバによって設定されます。

TPBroker では，インストール直後の監視時間の値は 180 秒です。Java アプリケーションでトランザクションを開始したときのデフォルトタイムアウト時間を変更するには，次に示すシステム環境定義で設定してください。

```
/OTS/TCS transaction_default_timeout
```

例

```
tsdefvalue /OTS/TCS transaction_default_timeout -i 32
```

複雑なトランザクション，大容量データベース，および高ネットワークトラフィックでは，大きな値の監視時間が適切です。ORB が提供する時間監視機能の詳細については，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

トランザクションコンテキストサーバは，アプリケーションプログラムがトランザクションを開始した時からトランザクションが決着するまでの経過時間を監視します。トランザクションは，トランザクションを開始したアプリケーションプログラムが，コミット要求またはロールバック要求のどちらかの決着リクエストを発行すると決着します。事前に設定された（ユーザが設定した）トランザクションの監視時間が過ぎると，トランザクションはロールバックされます。

ユーザはトランザクション開始時に，Current インタフェースの set_timeout オペレーションによって，トランザクションの監視時間を設定できます。

監視時間は，トランザクションコンテキストとともに，暗黙的にサーバアプリケーションにプロパゲーションされます。トランザクションコンテキストサーバはそれぞれのノードで監視を実行します。サーバノードは監視時間を，サーバがクライアントのトランザクショナルリクエストを受け取った時刻からトランザクションが決着するまでの間隔として解釈します。

トランザクションコンテキストサーバは，トランザクションコンテキスト管理方式が直接か間接かにかかわらず，監視を実行します。そのため，スレッドからサスペンドされたどのようなトランザクションでも，事前に設定された監視時間に到達するまでの間だけ監視します。

トランザクションの時間監視の詳細については，「8.　定義」を参照してください。

5.6　トランザクションマネージャへの接続

X/Open XA 準拠のリソースマネージャは，Java OTS とは接続できません。これは，XA インタフェースが Java 言語に適用できないためです。

OTS 仕様の Resource マネージャをサポートしているリソースマネージャは現在のところありません。したがって，DBMS などのリソースマネージャに接続するには，Java アプリケーションは C++ アプリケーションを経由して，リソースマネージャにアクセスする必要があります。

5.7 回復機能

Java OTS でも，システムの一部に障害が発生しても，部分回復処理をしてシステムが全面停止しないようにしたり，システムが全面停止しても，全面回復処理をしてシステムの状態を回復させたりできます。回復処理の詳細については「3.2　回復処理」を参照してください。ここでは，Java OTS で発生する，回復処理が機能しない障害のケースについて説明します。

5.7.1 トランザクションのタイムアウト

トランザクションコンテキストサーバは，アプリケーションプログラムがトランザクションを開始した時からトランザクションが決着するまでの経過時間を監視します。事前に設定されたトランザクションの監視時間に達すると，トランザクションコンテキストサーバはトランザクションを強制的にロールバックします。ほかのトランザクションコンテキストは，このロールバックオペレーションの影響を受けません。

5.7.2 ライトウェイト Java クライアントまたはサーバの異常終了

ライトウェイト Java クライアントまたはライトウェイト Java サーバが異常終了すると，トランザクションコンテキストサーバは，ClientProxy または ServerProxy オブジェクトに関連するすべてのトランザクションを，それぞれの監視時間が過ぎるとロールバックします。

5.7.3 トランザクションコンテキストサーバの異常終了

トランザクションコンテキストサーバが異常終了すると，Java OTS は障害を検出し，トランザクションコンテキストサーバの保持するすべてのトランザクションは自動的にロールバックされます。しかし，トランザクションがすでにコミットされていて，応答がまだ完了していない場合，トランザクションはコミットされます。

プロセス監視が使用されている場合，TPBroker は自動的にトランザクションコンテキストサーバを再開始します。

5.7.4 コミット時の障害

Java OTS を引き続き順調に動作させるために，オペレータが手動でトランザクションを決着させなければならない場合があります。C++ OTS では，コンテキストサーバ機能はクライアントプロセスまたはサーバプロセスの一部となっています。一方，Java OTS では，クライアントプロセスおよびサーバプロセスの両方がネットワークを介してトランザクションコンテキストサーバと通信します。Java OTS では，次に示す状況のどれか一つの状況が発生したらヒューリスティックエラーとなります。

- commit オペレーションの呼び出し後，Java クライアント（トランザクションオリジネータ）がクラッシュした。
- commit オペレーションの呼び出し後，Java クライアント（トランザクションオリジネータ）とトランザクションコンテキストサーバの間で通信障害が発生した。
- commit オペレーションの呼び出し後，トランザクションコンテキストサーバがクラッシュした。

上記のどれか一つが発生したら，トランザクションがコミットされたのかロールバックされたのかを調べてください。

オペレータがトランザクションを決着させるためには，トランザクションのグローバル ID（GID）が必要です。GID を調べるには，tslstrn コマンドを使用してください。トランザクションを決着するには，tscommit または tsrollback コマンドを使用し，GID を指定してください。

6 運用支援機能

この章では，TPBroker の運用支援機能（ADM）について説明します。

6.1　システム運用

TPBroker システムを運用するために必要な，システムの開始コマンドなどの各種運用コマンドを提供しています。運用コマンドについては，「9.　運用コマンド」を参照してください。

運用支援機能（ADM）を使用すると，OTS，ORB，アプリケーションプログラムなど，システムのさまざまなプロセスを一括起動，停止，および監視できます。

運用支援機能を使用するには，admsetup コマンドを使用して実行環境を OS に登録する必要があります。

admlaunch コマンドまたは admexec コマンドによる vbj コマンドおよびネーミングサービスの監視（Windows）

Windows XP，Windows 2003 対応版で vbj コマンド（起動する Java アプリケーションを含む）およびネーミングサービス（nameserv プロセス）を監視する場合で，コマンド列が 255 文字を超えるときは，admlaunch コマンドを使用します。
Windows Vista 対応版で vbj コマンド（起動する Java アプリケーションを含む）およびネーミングサービス（nameserv プロセス）を監視する場合で，コマンド列が 255 文字を超えるときは，admexec コマンドを使用します。admexec コマンドの使用方法は admlaunch コマンドと同一です。以下 admlaunch に統一して説明しますので，Windows Vista 対応版を御使用の場合 admlaunch を admexec に読み替えてください。
admlaunch コマンドは，vbj コマンドおよびネーミングサービスを監視するための専用のコマンドです。プロセス起動用コマンドおよびプロセス停止用コマンドの両方の用途で使用します。プロセス監視定義中に，または admstartprc，admstopprc コマンドの引数として使用します。admstartprc，admstopprc コマンドの詳細は，「9.3　運用コマンドの詳細」を参照してください。
起動コマンドとして動作する admlaunch コマンドは，運用監視機能の直接起動方式のプロセスとして使用します。また，admlaunch コマンドで起動した vbj コマンドおよびネーミングサービスは，必ず停止用コマンドとして動作する admlaunch コマンドを使用して停止させてください。admlaunch コマンドの詳細は，「9.3　運用コマンドの詳細」を参照してください。

6.2 プロセス監視

ここでは，運用支援機能の一つであるプロセス監視について説明します。

6.2.1 プロセス監視の概要

TPBroker では，運用支援機能の一つとして，プロセス監視機能を提供しています。TPBroker のプロセス監視機能には四つの方法があります。

プロセス監視の内容はプロセス監視定義ファイルで設定します。詳細は，「8.2　プロセス監視定義の詳細」を参照してください。

ここでは，すべてのプロセス監視方法に共通な機能を説明した上で，それぞれのプロセス監視方法について説明します。

(1) プロセス監視の共通機能

TPBroker では，プロセス監視として四つの方法を提供しています。四つの方法に共通な機能を次の表に示します。

表 6-1　プロセス監視の共通機能

項目	説明
監視対象となるプロセス	アプリケーションプログラム，リソースマネージャ，OTS や ORB のデーモンなど
最大監視対象プロセス数	運用定義 /ADM/max_process_num に設定された数
異常が発生した場合の処置	（UNIX） メッセージを syslog およびログファイルに出力する （Windows） メッセージをイベントログおよびログファイルに出力する

(2) プロセス監視方法

TPBroker のプロセス監視には，次に示す四つの方法があります。システム環境に応じてプロセス監視の方法を選択してください。

- 直接起動によるプロセス監視（直接起動方式）
  プロセス監視定義ファイルに監視対象プロセスを直接設定して監視する方法
- 間接起動によるプロセス監視（間接起動方式）
  プロセス監視定義ファイルに監視対象プロセス群を定義したコマンドを設定して監視する方法
- 運用コマンドによるプロセス監視
  運用コマンドを使用してプロセスを起動し，動的に監視対象に参加させる方法
- C++ の API によるプロセス監視

TPBroker が提供する C++ の API を使用して，手動で起動した C++ アプリケーションを動的に監視対象に参加させる方法。この方法は C++ OTS でだけ使用できます。

(3) 監視対象プロセスの出力情報

(UNIX)

監視対象プロセスの標準出力および標準エラー出力は自動的にファイルに切り替わります。出力先は，$ADMSPOOL/log/stdlog1 および stdlog2 です。環境変数 ADMSPOOL が未設定の場合は，$TPDIR/spool/log/stdlog1 および stdlog2 となります。

(Windows)

監視対象プロセスの標準出力および標準エラー出力は運用定義 /ADM/set_redirect_mode の設定値によってファイルに切り替わります。出力先は運用定義 /ADM/set_redirect_filename に従います。
ただし，C++ の API によるプロセス監視の場合は，監視対象プロセスの標準出力および標準エラー出力はファイルに切り替わることはありません。

監視対象プロセスの出力情報の注意事項を次に示します。

- 監視対象プロセスの標準出力および標準エラー出力は，パイプを経由してファイルに出力されます。パイプに対して高負荷での書き込みばかりが発生するとメッセージが破棄されることがあります。
- アプリケーションプログラムから printf などストリームを使用した出力を行う場合，出力前に出力ストリームバッファを切るか，出力後にストリームをフラッシュしてください。この処理を行わない場合，出力内容がファイルに反映されないことがあります。
- Windows 版の場合，監視対象プロセスの標準出力および標準エラー出力メッセージは，admtee.exe コマンドプロセスを経由して指定されたファイルに出力されます。そのため，オンライン中に admtee.exe プロセスが異常終了した場合，監視対象プロセスの標準出力および標準エラー出力メッセージは，ファイルに出力されなくなります。このような場合には，サービス「TPBroker」を停止し，再開始してください。
- Windows 版の場合，運用定義 /ADM/set_redirect_filename に設定されたファイルのオープン処理に失敗した場合（ディレクトリが存在しないなど），監視対象プロセスの標準出力および標準エラー出力先ファイルとして %ADMSPOOL%¥log¥stdlog1 と %ADMSPOOL%¥log¥stdlog2 が作成されます。

(4) 監視対象プロセスのカレントディレクトリ

(a) TPBroker の監視対象プロセスのカレントディレクトリの設定

TPBroker の監視対象プロセスのカレントディレクトリを次のように変更します。

- 環境変数 ADMSPOOL が設定済みの場合

$ADMSPOOL/tmp/home/" 識別子 "

● 環境変数 ADMSPOOL が未設定の場合
$TPDIR/spool/tmp/home/" 識別子 "

（UNIX）

監視対象プロセスが core を出力して異常終了した場合の，core ファイルの退避ディレクトリを次のようにします。

- 環境変数 ADMSPOOL が設定済みの場合
  $ADMSPOOL/errinfo/save
- 環境変数 ADMSPOOL が未設定の場合
  $TPDIR/spool/errinfo/save

監視対象プロセスの core ファイルは " 識別子 .N" という名称になります。ここで N は 1 から始まる通番で，三つまで退避します。通番が小さいほど，新しい core ファイルとなります。ただし，この機能は core ファイル名称が "core" である場合に限られます。

例

識別子が「OSAgent」の場合（環境変数 ADMSPOOL には /opt/TPBrokerV5/admspool が設定されているものとする）

- 監視対象プロセスのカレントディレクトリ
  /opt/TPBrokerV5/admspool/tmp/home/OSAgent
- 監視対象プロセスの core ファイルの退避ディレクトリ
  /opt/TPBrokerV5/admspool/errinfo/save
- 監視対象プロセスの退避した core ファイルの名称
  OSAgent.1，OSAgent.2，OSAgent.3

(b) 監視対象プロセスのカレントディレクトリが作成／削除されるタイミング

- admsetup -c 入力時
  $ADMSPOOL/tmp/home および $ADMSPOOL/errinfo/save を作成します。
  （UNIX）owner = root, group = sys, パーミッション = 777
- admsetup -d 入力時
  $ADMSPOOL/tmp/home および $ADMSPOOL/errinfo/save を削除します。
- 監視対象プロセスの起動時
  各プロセスのカレントディレクトリが作成されます。カレントディレクトリが作成できない場合，またはカレントディレクトリを移動できない場合は $ADMSPOOL がカレントディレクトリになります。
- ADM デーモンの起動時
  前回のセションで作成した各監視対象プロセスのカレントディレクトリを削除します。

(5) 作業ディレクトリの自動退避

TPBroker のプロセス監視機能は，環境変数 ADMSPOOL で設定された作業ディレクトリを使用します。

TPBroker のプロセス監視機能が運用中に異常終了した場合，次回起動時に自動的に前回使用した作業ディレクトリを退避します。退避するディレクトリ名は $ADMSPOOL_logN です。ここで N は 1，2，...，N となり，最大値は運用定義 /ADM/backup_count で設定された値になります。

例えば，環境変数 ADMSPOOL に /opt/TPBrokerV5/admspool を指定していた場合，退避ディレクトリは /opt/TPBrokerV5/admspool_log1 になります。

(6) プロセスのプロパティ設定

UNIX 版の TPBroker では，監視プロセスのプロパティ設定について，次の処理を行います。

(a) 設定するプロパティ

TPBroker のプロセス監視機能は以下のように監視プロセスのプロパティを設定します。

- 標準出力，標準エラー出力を，パイプに切り替えます。
- 現在のワークディレクトリを変更します。
- SIGPIPE シグナルを無視します。

HP-UX（IPF）版は 05-16-/C 以降，Solaris 版は 05-18-/A 以降，それ以外の UNIX プラットフォームでは 05-18 以降，admlaunchux コマンドを使用して SIGPIPE シグナルをデフォルトにしてプロセスを起動することができます。

(b) 設定できないプロパティ

TPBroker のプロセス監視機能は，以下のプロパティを設定する機能を持っていません。

- 補助グループ（2 次グループ）の ID

(c) 設定しないプロパティ

TPBroker のプロセス監視機能は，監視プロセスについて以下のプロパティを設定せず，OS の設定を引き継ぎます。

- プロセスグループ（ADMD（ADM デーモン）と同じプロセスグループとなります）
- ファイルモードの設定マスク（0 となります）
- SIGPIPE 以外のシグナルの設定

(7) ネットワークドライブアクセスに関する注意事項

Windows 版の場合，Windows の仕様により，サービスから起動されるプログラムからは，ネットワークドライブのアクセスに制限がありますので注意願います。

- 運用支援機能のプロセス監視定義に指定するプロセス名およびコマンド名は，ネットワークドライブ上のコマンドやプロセスを指定できません。
- 運用支援機能は，サービスから起動していますので，運用支援機能から起動されたプログラムは，ネットワークドライブにアクセスすることができません。各プログラム

はネットワークドライブをアクセスしないでください。

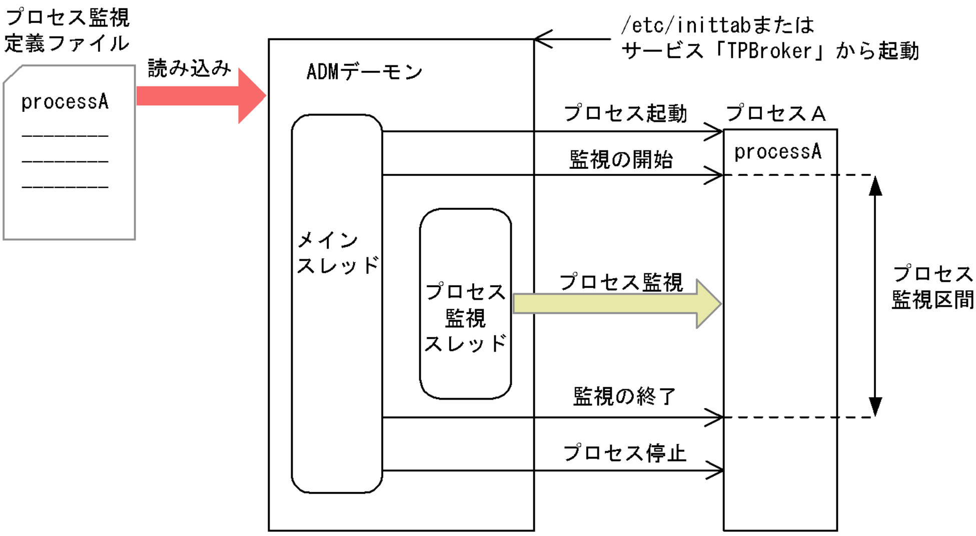
6.2.2　直接起動によるプロセス監視

この方法の場合，プロセス監視定義ファイルの設定およびプロセス停止用コマンドの作成が必要です。

(1)　監視方法

直接起動によるプロセス監視方法の概要を次の図に示します。

図 6-1　直接起動によるプロセス監視



- ADM デーモンがプロセス監視定義ファイルを読み込みます。
- 定義内容の解析とチェックを行います。
- 定義内容に沿ってプロセスを起動します。
- プロセスの監視を開始します。

(2)　監視対象プロセスの起動方法

TPBroker 開始時に渡されるプロセス監視定義ファイルを読み込み，そのプロセス監視定義ファイルで設定されたプロセスを起動します。プロセス監視定義ファイルは，TPBroker オンライン開始時に読み込まれます。

(3)　監視対象プロセスの起動のタイミング

次のどちらかをプロセス監視定義ファイルで選択できます。

- admstart コマンド入力時
  TPBroker オンライン開始時に同時に監視対象プロセスを起動します。
- admstartprc コマンド入力時

TPBroker オンライン開始後に任意に監視対象プロセスを起動します。

(4) 監視対象プロセスの起動成功の判断基準

起動したプロセスのプロセス ID を取得でき，プロセスの監視を開始できた場合に，起動成功と判断します。

(5) 監視対象プロセスの起動失敗時のアクション

プロセス監視定義ファイルに設定された監視対象プロセスに実行権限がないなどの理由でプロセス起動ができない場合，次のどれか一つをプロセス監視定義ファイルで選択できます。

- TPBroker の運用支援機能を終了させます。
  このとき，ほかの監視中のプロセスは停止させません。
- 起動に失敗したプロセスを再起動します。
  この場合，プロセス監視定義ファイルに設定された再起動用のプロセスを起動します。
- 何もしないで処理を続行します。

(6) 監視区間

admstart コマンドを発行してから admstop コマンドを発行するまで，または admstartprc コマンドを発行してから admstopprc コマンドを発行するまでの間です。

(7) 監視対象プロセス異常終了時のアクション

次のどれか一つをプロセス監視定義ファイルで選択できます。

- TPBroker の運用支援機能を終了させます。
- 異常終了したプロセスを再起動します。
- 何もしないで処理を続行します。
- コマンドなどのプロセスを起動します。
  このコマンドはプロセス監視定義ファイルでユーザが設定します。また，このプロセスは監視対象にはなりません。

TPBroker の運用支援機能を終了させた場合，ほかの監視中のプロセスは停止させません。再度 TPBroker が開始されたときに，再び監視対象に参加させます。

(8) 連続異常終了についての考慮

起動したプロセスが異常終了を繰り返す場合を考慮して，一定時間内（10 分間）に異常終了する回数の最大値をプロセス監視定義ファイルに設定できます。TPBroker は，異常終了検知後にそのプロセスを再起動することになっている場合でも，プロセスの連続異常終了回数が最大値に達しているときは，プロセスを再起動しません。

(9) 監視対象プロセスの停止方法

プロセス監視定義ファイルにプロセス停止用コマンドを設定している場合は，プロセス

停止用コマンドを起動します。プロセス停止用コマンドを設定していない場合は，システムコール（UNIX の場合は kill()，Windows の場合は TerminateProcess()）で直接プロセスを停止させます。

(10) 監視対象プロセスの停止のタイミング

次のどちらかをプロセス監視定義ファイルで選択できます。

- admstop コマンド入力時
  監視中の各プロセスを停止します。
- admstopprc コマンド入力時
  指定したプロセスを停止します。

(11) 直接起動によるプロセス監視機能が異常終了したときの考慮

ADM デーモンは Windows 版の場合は Windows のサービスとして，UNIX 版の場合は inittab に登録されています。ADM デーモンが異常終了した場合，ADM デーモンは自動的に再開始されます。

UNIX 版の場合，inittab から起動された ADM デーモンが 5 分間に 11 回連続で異常終了すると，プロセス監視機能は使用できなくなります。

監視対象プロセス異常終了時のアクションで「TPBroker の運用支援機能を終了させる」を選択した場合，そのほかの監視中のプロセスは停止されません。ADM デーモンが再開始したときに，再び監視対象プロセスに参加させます。この場合は，起動中の監視対象プロセスは ADM デーモンの子プロセスではありません。そのほかのプロセスも異常終了している場合は，プロセスを再起動します。

(12) プロセス停止用コマンド

プロセス停止用コマンドを作成するには，次に示す条件を満たす必要があります。

- プロセス停止用コマンド終了時には，監視対象プロセスが停止している。
- exit() による終了コードが「0」の場合は正常終了，「0」以外の場合は異常終了とする。

(13) プロセス強制停止用コマンド

プロセス強制停止用コマンドを作成するには，次に示す条件を満たす必要があります。

- プロセス強制停止用コマンド終了時には，監視対象プロセスが停止している。
- exit() による終了コードが「0」の場合は正常終了，「0」以外の場合は異常終了とする。
- 監視対象プロセスがない状態で実行しても問題ない。

(14) 注意事項

- プロセス監視定義ファイルに設定されたプロセス停止用コマンドで監視対象プロセスを停止できなかった場合，監視対象プロセスに対するプロセス監視を終了し，一定時

間（約 20 秒）経過後，監視対象プロセスをシステムコール※で強制停止させます。このとき，メッセージ KFCB29186-W がログファイルおよびシステムログに出力されます。

- プロセス監視定義ファイルに設定されたプロセス停止用コマンドの終了は，プロセス停止用コマンドのタイムアウト時間（プロセス停止用コマンドのタイムアウト値を設定しない場合は 5 分間）待ちます。それでも停止しない場合，ADM デーモンはコマンドプロセスをシステムコール※で強制停止させます。
- UNIX 版の場合，ADM デーモンは，監視対象プロセスやコマンドプロセスを kill() で停止させるときに，SIGTERM を使用します。SIGTERM で停止しない場合は，SIGKILL を使用して強制停止させます。
- UNIX 版の場合，プロセス監視定義ファイルに設定する監視対象プロセスやコマンドがシェルスクリプトのときは，必ず先頭の 2 バイトに文字「#!」を記述してください。「#!」を記述しない場合，シェルスクリプトの起動に失敗します。

注※

UNIX の場合のシステムコールは kill()，Windows の場合のシステムコールは TerminateProcess() です。

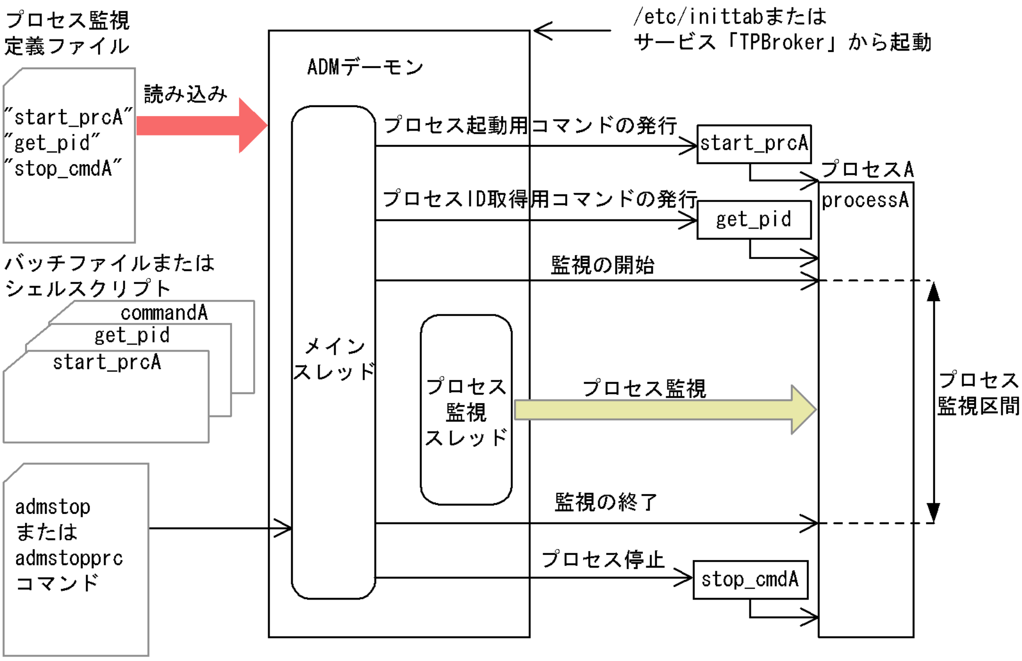
6.2.3 間接起動によるプロセス監視

この方法の場合，プロセス監視定義ファイルへの設定のほか，プロセス起動用，プロセス ID 取得用，およびプロセス停止用コマンド（バッチファイル，シェルスクリプトでも可）を用意する必要があります。

(1) 監視方法

間接起動によるプロセス監視方法の概要を次の図に示します。

図 6-2　間接起動によるプロセス監視



- ADM デーモンがプロセス監視定義ファイルを読み込みます。
- プロセス監視定義ファイルに設定されたプロセス起動用コマンドを発行します。
- プロセス監視定義ファイルに設定されたプロセス ID 取得用コマンドを発行します。
- 取得したプロセス ID を基に，監視対象に参加させます。

(2)　監視対象プロセスの起動方法

プロセス監視定義ファイルに設定されたプロセス起動用コマンドを基に監視対象プロセスを起動します。プロセス起動用コマンドは監視対象にはなりません。複数のプロセスが一つのプロセス起動用コマンドで起動することがあり，一つのプロセス起動用コマンドで起動したプロセスを，一つのプロセスグループとして扱います。

ユーザは，目的の監視対象プロセスを起動するプロセス起動用コマンドをあらかじめ用意する必要があります。

(3)　監視対象プロセスの起動のタイミング

次のどちらかをプロセス監視定義ファイルで選択できます。

- admstart コマンド入力時
  TPBroker オンライン開始時に同時に監視対象プロセスを起動します。
- admstartprc コマンド入力時
  TPBroker オンライン開始後に任意に監視対象プロセスを起動します。

(4) 監視対象プロセスの起動成功の判断基準

プロセス ID 取得用コマンドを発行し，プロセス ID を取得できた場合に起動成功と判断します。このコマンドが正常終了して取得できた複数のプロセスをプロセスグループとして扱います。

ユーザは，プロセス ID 取得用コマンドをあらかじめ用意しておく必要があります。プロセス ID 取得用コマンドがプロセス監視定義ファイルに設定されていない場合は，プロセス起動用コマンドとプロセス停止用コマンドによるプロセスの起動および停止だけを行い，プロセスの監視はしません。

(5) 監視対象プロセスの起動失敗時のアクション

プロセス監視定義ファイルに設定された監視対象プロセスに実行権限がないなどの理由でプロセスを起動できない場合，次のどれか一つをプロセス監視定義ファイルで選択できます。

- TPBroker の運用支援機能を終了させます。
- 起動に失敗したプロセスを再起動します。
- 何もしないで処理を続行します。

(6) 監視区間

プロセス監視定義ファイルに設定したプロセス ID 取得用コマンドでプロセス ID を取得してから，プロセス監視定義ファイルに設定したプロセス停止用コマンドを発行するまでの間です。

(7) 監視対象プロセス異常終了時のアクション

次のどれか一つをプロセス監視定義ファイルで選択できます。

- TPBroker の運用支援機能を終了させます。
- 異常終了したプロセスを再起動します。
- 何もしないで処理を続行します。
- コマンドなどのプロセスを起動します。
  このコマンドはプロセス監視定義ファイルでユーザが設定します。また，このプロセスは監視対象にはなりません。

プロセスが異常終了した場合，そのプロセスと同一のプロセスグループ内のすべてのプロセスを監視対象から外し，停止します。

ここで，プロセスグループに１つのプロセスしか含まれない場合でも，以下のアクションが発生します。

- 強制停止コマンドが定義されている場合，強制停止コマンドを実行します。
- 強制停止コマンドが定義されていない場合，かつ正常停止コマンドが定義されている場合，正常停止コマンドを実行します。
- 強制停止コマンド，正常停止コマンド共に定義されていない場合，プロセスグループ

内のダウンした以外のプロセスをシステムコール（UNIX の場合は kill()，Windows の場合は TerminateProcess()）で直接プロセスを停止します。

(8) 連続異常終了についての考慮

起動したプロセスが異常終了を繰り返す場合を考慮して，一定時間内（10 分間）に異常終了する回数の最大値をプロセス監視定義ファイルに設定できます。TPBroker は，異常終了検知後にそのプロセスを再起動することになっている場合でも，プロセスの連続異常終了回数が最大値に達しているときは，プロセスを再起動しません。

(9) 監視対象プロセスの停止方法

プロセス監視定義ファイルにプロセス停止用コマンドを設定している場合は，プロセス停止用コマンドを起動します。プロセス停止用コマンドを設定していない場合は，システムコール（UNIX の場合は kill()，Windows の場合は TerminateProcess()）で直接プロセスを停止させます。

(10) 監視対象プロセスの停止のタイミング

次のどちらかをプロセス監視定義ファイルで選択できます。

- admstop コマンド入力時
  監視中の各プロセスを停止します。
- admstopprc コマンド入力時
  指定したプロセスを停止します。

(11) 間接起動によるプロセス監視機能が異常終了したときの考慮

ADM デーモンは Windows 版の場合は Windows のサービスとして，UNIX 版の場合は inittab に登録されています。ADM デーモンが異常終了した場合，ADM デーモンは自動的に再開始されます。

UNIX 版の場合，inittab から起動された ADM デーモンが 5 分間に 11 回連続で異常終了すると，プロセス監視機能は使用できなくなります。

監視対象プロセス異常終了時のアクションで「TPBroker の運用支援機能を終了させる」を選択した場合，そのほかの監視中のプロセスは停止されません。ADM デーモンが再開始したときに，再び監視対象プロセスに参加させます。この場合は，起動中の監視対象プロセスは ADM デーモンの子プロセスではありません。そのほかのプロセスも異常終了している場合は，プロセスを再起動します。

(12) プロセス起動用コマンド

プロセス起動用コマンドを作成するには，次に示す条件を満たす必要があります。

- プロセス起動用コマンドの終了時には，プロセス停止用コマンドが実行可能状態になっている。
- プロセス起動用コマンドの終了時には，全監視対象プロセスが起動完了している。

- exit() による終了コードが「0」の場合は正常終了,「0」以外の場合は異常終了とする。

ADM デーモンでは，指定されたプロセス起動用コマンドの終了を待ちます。

(13) プロセス停止用コマンド

プロセス停止用コマンドを作成するには，次に示す条件を満たす必要があります。

- プロセス停止用コマンドの終了時には，プロセス起動用コマンドが実行可能状態になっている。
- プロセス停止用コマンドの終了時には，プロセス起動用コマンドで起動したプロセスが停止している。
- exit() による終了コードが「0」の場合は正常終了,「0」以外の場合は異常終了とする。
- 同一の識別子で監視対象になっているプロセスだけを停止する。

(14) プロセス強制停止用コマンド

プロセス強制停止用コマンドを作成するには，次に示す条件を満たす必要があります。

- プロセス強制停止用コマンドの終了時には，プロセス起動用コマンドが実行可能状態になっている。
- exit() による終了コードが「0」の場合は正常終了,「0」以外の場合は異常終了とする。
- 監視対象プロセスがない状態で実行しても問題ない。
- 同一の識別子で監視対象になっているプロセスだけを停止する。

(15) プロセス ID 取得用コマンド

プロセス ID 取得用コマンドを作成するには，次に示す条件を満たす必要があります。次の条件を満たしていないと，監視対象プロセスのプロセス ID を取得できなくなり，正常にプロセス監視ができない場合があります。

- プロセス起動用コマンドで起動したプロセスのプロセス ID を標準出力に出力する。
- exit() による終了コードが「0」の場合は正常終了,「0」以外の場合は異常終了とする。
- 出力フォーマットは次のようにする。

9	9	9	6	¥n	
1	0	0	2	3	¥n
1	0	2	5	7	¥n

各行の最後には「¥n」が付くようにします。

(16) 注意事項

- プロセス ID 取得用コマンドが未設定の場合は，プロセスの監視は行いません。また,

この場合，プロセス停止用コマンドは必ず指定してください。

- プロセス ID 取得用コマンドの指定がある場合は，プロセス停止用コマンドの指定がなくてもかまいません。ただし，この場合はシステムコール※で直接プロセスを停止させます。
- 間接起動によるプロセス監視によって，一つの識別子で 100 個までのプロセスを監視できます。プロセス ID 取得用コマンドを発行した結果，100 個を超えるプロセス ID を取得した場合，メッセージを出力し起動したプロセスを停止します。停止は，強制正常停止用コマンドを発行します。強制正常停止用コマンドが設定されていない場合は，システムコール※で直接プロセスを停止させます。
- 間接起動によってプロセスを起動および監視するときに，監視対象とするプロセスと同じ名前のプロセスがすでに存在する場合，プロセス ID 取得用コマンドの仕様によっては，すでに起動されているプロセスも監視対象になることがあります。プロセス ID 取得用コマンドでは，間接起動によって起動したプロセスのプロセス ID だけを取得するようにしてください。
- プロセス監視定義ファイルに設定されたプロセス停止用コマンドで監視対象プロセスを停止できなかった場合，監視対象プロセスに対するプロセス監視を終了し，一定時間（約 20 秒）経過後，監視対象プロセスをシステムコール※で強制終了させます。このとき，メッセージ KFCB29186-W がログファイルおよびシステムログに出力されます。
- プロセス監視定義ファイルに設定されたプロセス ID 取得用コマンドおよびプロセス停止用コマンドの終了は，プロセス停止用コマンドのタイムアウト時間（プロセス停止用コマンドのタイムアウト値を設定しない場合は 5 分間）待ちます。それでも停止しない場合，コマンドプロセスをシステムコール※で強制停止させます。
- UNIX 版の場合，監視対象プロセスやコマンドプロセスを kill() システムコールで停止させるときには，SIGTERM を使用します。SIGTERM で停止しない場合は，SIGKILL を使用して強制停止させます。
- UNIX 版の場合，プロセス監視定義ファイルに設定する監視対象プロセスやコマンドがシェルスクリプトのときは，必ず先頭の 2 バイトに文字「#!」を記述してください。「#!」を記述しない場合，シェルスクリプトの起動に失敗します。

注※

UNIX の場合のシステムコールは kill()，Windows の場合のシステムコールは TerminateProcess() です。

6.2.4 運用コマンドによるプロセス監視

この方法では，「6.2.2 直接起動によるプロセス監視」で示したプロセス監視定義ファイルに設定されたプロセスを起動して監視対象とする場合と，プロセス監視定義ファイルに設定されていないプロセスを起動して監視対象とする場合があります。

注

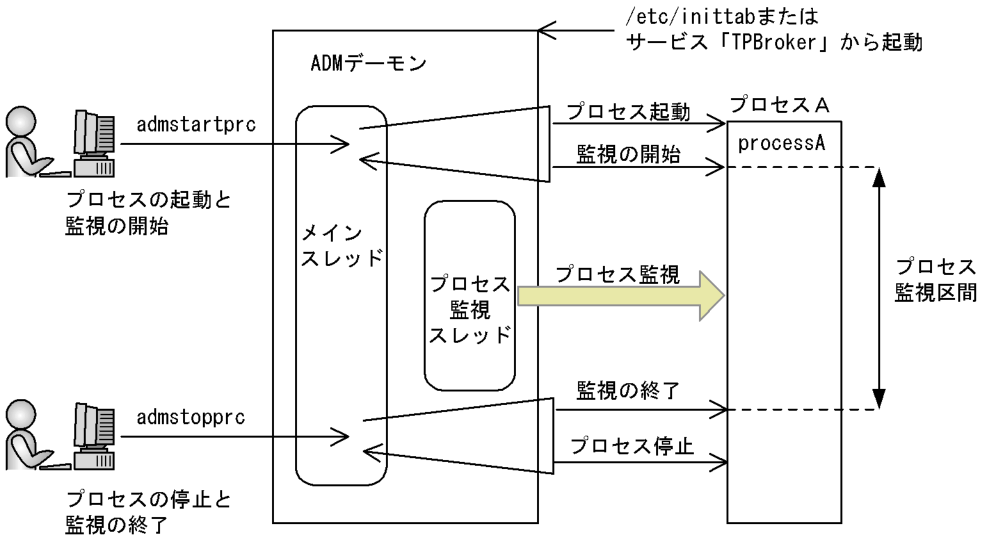
この方法では，プロセス監視定義ファイルに設定されていない間接起動方式のプロ

セスを起動および監視することはできません。

(1) 監視方法

運用コマンドによるプロセス監視方法の概要を次の図に示します。

図 6-3 運用コマンドによるプロセス監視



- admstartprc コマンドを入力します。
- 指定されたプロセスがプロセス監視定義ファイルに設定されているかチェックします。
- プロセスを起動します。
- プロセスの監視を開始します。

(2) 監視対象プロセスの起動方法

TPBroker が提供する admstartprc コマンドを入力し，TPBroker に対してプロセスの起動を依頼し , プロセスを起動します。

(3) 監視対象プロセスの起動のタイミング

admstartprc コマンドが入力されたときです。

(4) 監視対象プロセスの起動成功の判断基準

起動したプロセスのプロセス ID を取得でき，プロセスの監視を開始できた場合に，起動成功と判断します。

(5) 監視対象プロセスの起動失敗時のアクション

admstartprc コマンドを使用した場合，プロセス監視定義ファイルにすでに設定されているプロセスとそうでないプロセスを起動できます。

すでにプロセス監視定義ファイルに設定されているプロセスの起動に失敗した場合は，次のどれか一つをプロセス監視定義ファイルで選択できます。

- TPBroker の運用支援機能を終了させます。
- 起動に失敗したプロセスを再起動します。
- 何もしないで処理を続行します。

プロセス監視定義ファイルに設定されていないプロセスの起動に失敗した場合は，admstartprc コマンドのオプションで次のどちらかを選択できます。

- TPBroker をダウンさせます（-f オプション）。
- 何もしないで処理を続行します（デフォルト）。

(6) 監視区間

admstart コマンドを発行してから admstop コマンドを発行するまで，または admstartprc コマンドを発行してから admstopprc コマンドを発行するまでの間です。

(7) 監視対象プロセス異常終了時のアクション

プロセス監視定義ファイルに設定されたプロセスが起動または監視時に異常終了した場合は，次のどれか一つをプロセス監視定義ファイルで選択できます。

- TPBroker の運用支援機能を終了させます。
- 異常終了したプロセスを再起動します。
- 何もしないで処理を続行します。
- コマンドなどのプロセスを起動します。
  このコマンドはプロセス監視定義ファイルでユーザが設定します。

プロセス監視定義ファイルに設定されていないプロセスが起動または監視時に異常終了した場合は，admstartprc コマンドのオプションで次のどちらかを選択できます。

- TPBroker をダウンさせます（-f オプション）。
- 何もしないで処理を続行します（デフォルト）。

(8) 連続異常終了についての考慮

プロセス監視定義ファイルに指定されたプロセスの監視では，起動したプロセスが異常終了を繰り返す場合を考慮して，一定時間内（10 分間）に異常終了する回数の最大値をプロセス監視定義ファイルに設定できます。TPBroker は，異常終了検知後にそのプロセスを再起動することになっている場合でも，プロセスの連続異常終了回数が最大値に達しているときは，プロセスを再起動しません。

プロセス監視定義ファイルに指定されていないプロセスの場合，異常終了しても再起動されないため，連続異常終了は起こりません。

(9) 監視対象プロセスの停止方法

プロセス監視定義ファイルに指定されたプロセスを起動または監視した場合は，

「6.2.2(9)　監視対象プロセスの停止方法」で示した方法で停止させます。

プロセス監視定義ファイルに指定されていないプロセスを起動または監視した場合は，admstopprc コマンドの -o オプションで指定されたプロセス停止用コマンドを TPBroker から発行して停止させます。このオプションに指定がない場合は，システムコール（UNIX の場合は kill()，Windows の場合は TerminateProcess()）で直接プロセスを停止させます。admstopprc コマンドの -o オプションで指定されたプロセス停止用コマンドは，次に示す条件を満たす必要があります。

- プロセス停止用コマンド終了時には，監視対象プロセスが停止している。
- exit() による終了コードが「0」の場合は正常終了，「0」以外の場合は異常終了とする。

(10) 監視対象プロセスの停止のタイミング

TPBroker が提供する admstopprc コマンドで，監視中のプロセスを停止させます。また，admstop コマンドで TPBroker を終了させる場合は次のようにします。

- プロセス監視定義ファイルに設定されたプロセスの場合は，定義の設定によって監視中のプロセスを停止させます。
- プロセス監視定義ファイルに設定されていないプロセスの場合は，プロセス停止用コマンドが未設定であるため，システムコール（UNIX の場合は kill()，Windows の場合は TerminateProcess()）で直接プロセスを停止させます。

(11) 運用コマンドによるプロセス監視機能が異常終了したときの考慮

ADM デーモンは Windows 版の場合は Windows のサービスとして，UNIX 版の場合は inittab に登録されています。ADM デーモンが異常終了した場合，ADM デーモンは自動的に再開始されます。

UNIX 版の場合，inittab から起動された ADM デーモンが 5 分間に 11 回連続で異常終了すると，プロセス監視機能は使用できなくなります。

監視対象プロセス異常終了時のアクションで「TPBroker の運用支援機能を終了させる」を選択した場合，そのほかの監視中のプロセスは停止されません。ADM デーモンが再開始したときに，再び監視対象プロセスに参加させます。この場合は，起動中の監視対象プロセスは ADM デーモンの子プロセスではありません。そのほかのプロセスも異常終了している場合は，プロセスを再起動します。

(12) 注意事項

- admstopprc コマンドの -o オプションの引数で指定されたプロセス停止用コマンドで監視対象プロセスを停止できなかった場合，監視対象プロセスに対するプロセス監視を終了し，一定時間（約 20 秒）経過後，監視対象プロセスをシステムコール※で強制停止させます。このとき，メッセージ KFCB29186-W がログファイルおよびシステムログに出力されます。
- プロセス ID 取得用コマンドおよびプロセス停止用コマンドの終了は，5 分間待ちま

す。それでも停止しない場合，コマンドプロセスをシステムコール※で強制停止させます。
- UNIX 版の場合，監視対象プロセスやコマンドプロセスを kill() システムコールで停止させるときには，SIGTERM を使用します。SIGTERM で停止しない場合は，SIGKILL を使用して強制停止させます。
- UNIX 版の場合，プロセス監視定義ファイルに設定する監視対象プロセスやコマンド，admstartprc コマンドおよび admstopprc コマンドの -o オプションの引数で指定する監視対象プロセスやコマンドがシェルスクリプトのときは，必ず先頭の 2 バイトに文字「#!」を記述してください。「#!」を記述しない場合，シェルスクリプトの起動に失敗します。

注※

UNIX の場合のシステムコールは kill()，Windows の場合のシステムコールは TerminateProcess() です。

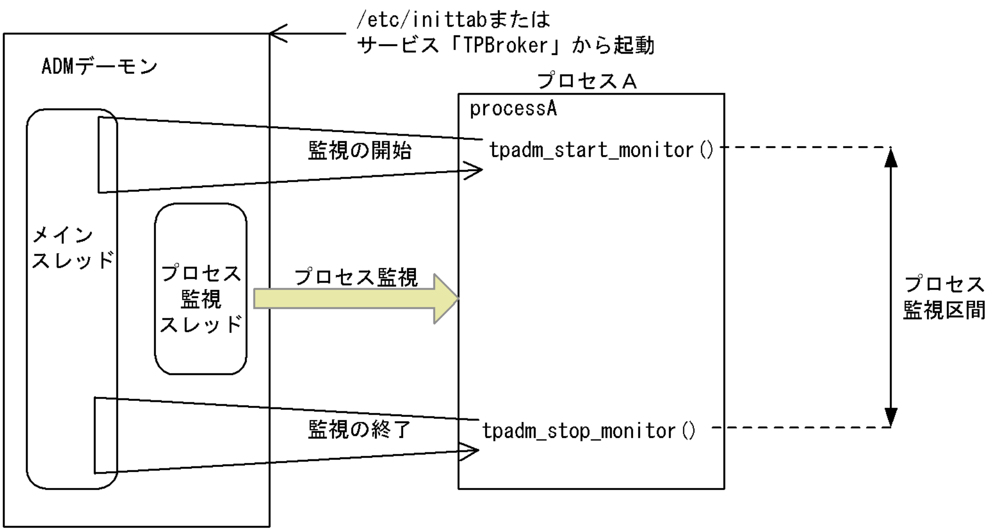
6.2.5　C++ の API によるプロセス監視

C++ の API によるプロセス監視方法を説明します。TPBroker の運用支援機能を使用する場合のアプリケーションプログラムインタフェースの文法については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

(1)　監視方法

C++ の API によるプロセス監視方法の概要を次の図に示します。

図 6-4　C++ の API によるプロセス監視



C++ の API を使用して動的にプロセス監視に参加できる C++ アプリケーションを，

admstartprc コマンドを使用して動的に参加させようとした場合，admstartprc コマンドによるプロセス監視を優先します。

(2) 監視対象プロセスの起動方法

ユーザが手動で起動します。

(3) 監視対象プロセスの起動のタイミング

起動は任意です（ただし，TPBroker が稼働している間です）。

(4) 監視対象プロセスの起動成功の判断基準

TPBroker の外部からの起動のため，ありません。

(5) 監視対象プロセスの起動失敗時のアクション

TPBroker の外部からの起動のため，ありません。

(6) 監視区間

C++ アプリケーション中で tpadm_start_monitor() を呼び出してから tpadm_stop_monitor() を呼び出すまでの間です。

(7) 監視対象プロセス異常終了時のアクション

メッセージを出力します。また，tpadm_start_monitor() の引数で次のどちらかを選択できます。

- TPBroker の運用支援機能を終了させます（TPADM_DOWN を指定）。
- 何もしないで処理を続行します（TPADM_NOFLAGS を指定）。

(8) C++ の API によるプロセス監視機能が異常終了したときの考慮

手動で起動したプロセスの場合，TPBroker がダウンしてもプロセスは停止させません。TPBroker が稼働していない状態で，tpadm_stop_monitor() を呼び出すと，この API はエラーリターンします。

TPBroker がダウンしたあと再開始されると，前回のセション時に監視中だったプロセスを再度監視対象に参加させます。

6.3 監視対象プロセス並列起動／停止機能

監視対象プロセスの起動および停止を並列で行う機能について説明します。

6.3.1 概要

TPBroker 開始，終了，および再開始時の監視対象プロセス並列起動／停止機能は，admstart コマンド入力時，admstop コマンド入力時，および TPBroker（ADM）ダウン後の再開始時に，ユーザがプロセス監視定義ファイルに設定したグループごとに ADM の監視対象プロセスを並列に起動または停止する機能です。

この機能を有効にするためには，システム環境定義の追加とプロセス監視定義ファイルの編集が必要です。

6.3.2 システム環境定義

監視対象プロセス並列起動／停止機能を使用するには，運用定義 /ADM/set_parallel_mode および /ADM/set_parallel_count を設定する必要があります。それぞれの運用定義の詳細は「8.3.1　運用定義」を参照してください。

6.3.3 プロセス監視定義ファイル

監視対象プロセス並列起動／停止機能を使用するには，プロセス監視定義ファイルの起動プロセスの識別子を編集する必要があります。

(1) 形式

識別子 ,order= 順序

(2) 説明

プロセス監視定義ファイル内の識別子設定領域に，起動順序を設定します。

この指定がない場合はシリアルで起動します。「識別子 <u>,order= 順序</u>」の下線の部分には空白，またはタブを記述しないでください。また，順序の個所に数字以外の文字を記述しないでください。記述した場合は，順序未設定と判断し，エラーメッセージを出力します。プロセス監視定義ファイル内の識別子の詳細については，「8.2　プロセス監視定義の詳細」を参照してください。

「順序」に指定できる数値は 1 ～ 4096 です。それ以外は順序未設定と判断し，エラーメッセージを出力します。また間を空けて（1 の次に 2 がなくて 3 が定義されているような場合）設定することもできます。

(3) 記述例

```
OSAgent,order=1:"/opt/TPBrokerV5/bin/osagent -g":～
0001,order=2:"/opt/TPBrokerV5/bin/tsstart":～
testap01,order=3:"/usr/home/user1/test/ap01 -OAlocalipc 1":～
testap02,order=3:"/usr/home/user1/test/ap02 -OAlocalipc 1":～
..............
```

6.3.4 プロセス起動順序

admstart コマンド入力時のプロセス起動順序について説明します。

(1) 運用定義 /ADM/set_parallel_mode が "N" または未設定の場合

プロセス監視定義ファイルに設定された順に，シリアルに起動します。

(2) 運用定義 /ADM/set_parallel_mode が "Y" の場合

プロセス監視定義ファイルに設定された「order=XXX」に従って，順次起動します。同じ order は並列に起動します。

なお，プロセス監視定義ファイルに「order=XXX」が設定されていないプロセスは，すべてのプロセス起動後にシリアルに起動します。したがって，運用定義 /ADM/set_parallel_mode が "Y" の場合でも，プロセス監視定義ファイルに「order=XXX」がないときは，(1) と同じ動作をします。また，「order=XXX」が設定されていても，起動のタイミングが「command」（admstartprc コマンドによる起動）指定の場合は，admstart コマンド入力時には起動しません。

なお，order で指定される同一グループ内での起動順序は不定です。

定義例に従って，運用定義 /ADM/set_parallel_mode が "Y" の場合の起動順序を説明します。

● **定義例** 1

```
OSAgent,order=1:"/opt/TPBrokerV5/bin/osagent -g":～
0001,order=2:"/opt/TPBrokerV5/bin/tsstart":～
testap01,order=3:"/usr/home/user1/test/ap01 -OAlocalipc 1":～
testap02,order=3:"/usr/home/user1/test/ap02 -OAlocalipc 1":～.
.............
```

定義例 1 の起動順序は OSAgent，0001 の順に起動したあと，testap01 と testap02 を並列に起動します。

● **定義例** 2

```
OSAgent,order=1:"/opt/TPBrokerV5/bin/osagent -g":～
0001,order=2:"/opt/TPBrokerV5/bin/tsstart":～
testap01:"/usr/home/user1/test/ap01 -OAlocalipc 1":～
testap02,order=3:"/usr/home/user1/test/ap02 -OAlocalipc 1":～
testap03,order=3:"/usr/home/user1/test/ap03 -OAlocalipc 1":～
..............
```

定義例 2 の起動順序は OSAgent，0001 の順に起動したあと，testap02 と testap03 を並列に起動し，その後 testap01 を起動します。

● **定義例** 3

```
OSAgent,order=1:"/opt/TPBrokerV5/bin/osagent -g":～
0001,order=2:"/opt/TPBrokerV5/bin/tsstart":～
testap01,order=3:"/usr/home/user1/test/ap01 -OAlocalipc 1"
                  :～...:command:～
testap02,order=3:"/usr/home/user1/test/ap02 -OAlocalipc 1"
                  :～...:none:～
testap03,order=3:"/usr/home/user1/test/ap03 -OAlocalipc 1"
                  :～...:none:～
..............
```

定義例 3 の起動順序は OSAgent，0001 の順に起動したあと，testap02 と testap03 を並列に起動します。testap01 は admstart コマンド入力時には起動しないで，admstartprc コマンド入力時に起動します。

6.3.5　プロセス停止順序

admstop コマンド入力時のプロセス停止順序について説明します。

(1)　運用定義 /ADM/set_parallel_mode が "N" または未設定の場合

起動順序の逆順でプロセスをシリアルに停止します。

(2)　運用定義 /ADM/set_parallel_mode が "Y" の場合

プロセス監視定義ファイルに設定された「order=xxx」に従って，order の逆順にグループごとに並列に停止します。なお，admstartprc コマンドで起動したプロセス監視定義ファイルに設定されていないプロセスや，プロセス監視定義ファイルに起動順序である「order=xxx」が設定されていないプロセスに関しては，システム停止時の最初にプロセス起動順の逆順にシリアルに停止します。したがって，運用定義 /ADM/set_parallel_mode が "Y" の場合でも，プロセス監視定義ファイルに「order=xxx」がないときは，(1) と同じ動作をします。

なお，order で指定される同一グループ内での停止順序は不定です。

定義例に従って，運用定義 /ADM/set_parallel_mode が "Y" の場合の停止順序を説明します。

● **定義例** 1

```
OSAgent,order=1:"/opt/TPBrokerV5/bin/osagent -g":～
0001,order=2:"/opt/TPBrokerV5/bin/tsstart":～
testap01:"/usr/home/user1/test/ap01 -OAlocalipc 1":～
```

```
testap02,order=3:"/usr/home/user1/test/ap02 -OAlocalipc 1":～
testap03,order=3:"/usr/home/user1/test/ap03 -OAlocalipc 1":～
...............
```

定義例 1 のようにプロセス起動順序が，OSAgent → 0001 → testap02,testap03 → testap01 → testap04（admstartprc コマンドで起動した未設定のプロセス）の場合は，admstop コマンド入力時のプロセスの停止順序が testap04 → testap01 → testap02,testap03 → 0001 → OSAgent となります。

6.3.6 システム再開始時のプロセス起動順序

TPBroker（ADM）が異常終了したあと，監視対象プロセスを再起動する場合，または -fr オプションを指定した admstop コマンドで TPBroker を終了したあと，admstart コマンドで再起動する場合の起動順序について説明します。

（1） 運用定義 /ADM/set_parallel_mode が "N" または未設定の場合

前回の監視状態を基に，起動順にプロセスを再起動します。なお，再起動時に前回監視していたプロセスが生存していた場合は，そのプロセスをそのまま監視対象プロセスに含めます。

（2） 運用定義 /ADM/set_parallel_mode が "Y" の場合

前回の監視状態を基に，プロセス監視定義に設定されている「order=xxx」に従って，グループごとに並列に起動します。なお，admstartprc コマンドで起動したプロセス監視定義ファイルに設定されていないプロセスや，プロセス監視定義ファイルに起動順序である「order=xxx」が設定されていないプロセスに関しては，システム再開始時の最後にプロセス起動順にシリアルに起動します。

なお，order で指定される同一グループ内での起動順序は不定です。

定義例に従って，運用定義 /ADM/set_parallel_mode が "Y" の場合の起動順序を説明します。

● **定義例** 1

```
OSAgent,order=1:"/opt/TPBrokerV5/bin/osagent -g":～
0001,order=2:"/opt/TPBrokerV5/bin/tsstart":～
testap01:"/usr/home/user1/test/ap01 -OAlocalipc 1":～
testap02,order=3:"/usr/home/user1/test/ap02 -OAlocalipc 1":～
testap03,order=3:"/usr/home/user1/test/ap03 -OAlocalipc 1":～
...............
```

定義例 1 で，プロセス起動順序が，OSAgent → 0001 → testap02，testap03 → testap01 → testap04（admstartprc コマンドで起動した未設定のプロセス）の場合は，システム再開始時のプロセスの起動順序が OSAgent → 0001 → testap02，testap03 → testap01 → testap04 となります。

6.3.7　admstartprc，admstopprc，および admreload コマンド実行時の考慮

admstartprc，admstopprc，および admreload コマンドを使用してプロセスを起動または停止する場合の注意について説明します。

(1) admstartprc コマンドで複数のプロセスを起動する場合

admstartprc コマンドで複数のプロセスを起動する場合，次に示す動作をします。

(a) -p オプションが指定されない場合

プロセス監視定義ファイルに order が設定されていても，すべてシリアルに起動します。

(b) -p オプションが指定された場合

プロセス監視定義ファイルに order が設定されていても，すべて並列に起動します。したがって，異なる order が指定されているプロセスでも，同時に起動することになります。指定されたプロセスの定義を読み込んで，order ごとに並列に実行する処理は行いません。

プロセス起動のタイミングが「command」（admstartprc コマンドによる起動）の場合，プロセス監視定義ファイルに「order=XXX」が設定されていても，order の設定は起動時には無効になります。

(2) admstopprc コマンドで複数のプロセスを停止する場合

admstopprc コマンドで複数のプロセスを停止する場合，次に示す動作をします。

(a) -p オプションが指定されない場合

プロセス監視定義ファイルに order が設定されていても，すべてシリアルに停止します。

(b) -p オプションが指定された場合

プロセス監視定義ファイルに order が設定されていても，すべて並列に停止します。したがって，異なる order が設定されているプロセスでも，同時に停止することになります。指定されたプロセスの定義を読み込んで，order ごとに並列に実行する処理は行いません。

(3) admreload コマンドで削除した定義のプロセスを停止する場合

admreload コマンドの -f オプションを指定した場合，削除したプロセス監視定義に対応するプロセスが生存していれば，そのプロセスを停止します。このときの処理は，運用定義 /ADM/set_parallel_mode が "Y" の場合でも，プロセス起動順の逆順にシリアルに停止します。

(4) admreload コマンドで変更した定義のプロセスを停止／起動する場合

admreload コマンドの -f オプションを指定した場合，変更したプロセス監視定義に対応するプロセスが生存していれば，そのプロセスを停止したあとに再起動します。このときの処理は，運用定義 /ADM/set_parallel_mode が "Y" の場合でも，プロセスの停止と再起動をシリアルに行います。

また，次のように，プロセス監視定義ファイル内の order を変更した場合，admreload コマンドでは「変更」として扱います。

（変更前）

```
OSAgent,order=1:～
     ↓
```

（変更後）

```
OSAgent,order=2:～........
```

このときは，admreload コマンドの引数には，「OSAgent」だけを指定します。「OSAgent,order=1」を指定した場合は，エラーとなります。

● 出力例

```
% admreload -i OSAgent -l
changed:OSAgent
ORDER:
before:1
after:2
```

6.4 プロセス監視定義ファイルの再読み込み機能

ADM のプロセス監視定義ファイルの再読み込み機能とは，admstart コマンド入力後，またはシステムダウン後の再開始によって，プロセス監視を行っている状態から，admstop コマンド入力までの間にプロセス監視定義を変更する機能です。この機能によって TPBroker を終了することなく，監視対象プロセスの定義の変更を反映できます。

6.4.1 プロセス監視定義ファイルの再読み込み機能の概要

プロセス監視定義ファイルの再読み込み機能は admreload コマンドで提供します。admsetup コマンド入力時に指定したプロセス監視定義ファイルを編集したあと，admreload コマンドを入力します。ADM は，プロセス管理テーブルに格納してある情報と admreload コマンド入力時のプロセス監視定義ファイルの内容を比較し，内容が削除，変更，または追加されている場合，プロセス監視定義ファイルの内容をプロセス管理テーブルに反映します。admreload コマンド入力時のプロセス管理テーブルとプロセス監視定義ファイルの内容を比較するので，複数回実行できます。

定義を比較する順序（プロセス起動または停止の順序）は，削除→変更→追加の順です。途中，異常定義があった場合も全件比較します。書き換えられる定義はすべて書き換えます。定義の削除，変更，および追加に関する各動作について次に示します。

(1) 定義削除の動作について

(a) プロセスが停止中の場合

プロセス管理テーブルを削除します。削除は即時有効となります。

(b) プロセスが起動中の場合

削除するかどうかは，admreload コマンドのオプションで指定できます。プロセスが起動中の場合の定義削除のオプションの仕様について次の表に示します。

表 6-2 定義削除のオプションの仕様

オプション	オプションの仕様
なし	1. 警告メッセージを出力する。 2. プロセス管理テーブルは書き換えない。 修正された定義は，admreload コマンドを -f オプション付きで再度入力するか，または次回 ADM 正常開始もしくは強制正常開始するまで反映されない。
-f	1. プロセス管理テーブルを書き換える。 2. 起動中のプロセスを，削除前の定義に従って停止させる。

(2) 定義変更の動作について

変更後の定義にエラーがある場合は，プロセス管理テーブルは書き換えられないで変更前の状態のままになります。

(a) プロセスが停止中の場合

プロセス管理テーブルを書き換えますが，プロセス起動はしません。

(b) プロセスが起動中の場合

変更するかどうかは，admreload コマンドのオプションで指定できます。このオプションの仕様は，変更するときの状況によって異なります。

変更するときの状況には，前回のプロセス監視定義ファイルからの変更の場合，admstartprc コマンドによる動的監視プロセスと同じ識別子をプロセス監視定義ファイルに設定した場合，および API（tpadm_start_monitor()）による動的監視プロセスと同じ識別子をプロセス監視定義ファイルに設定した場合の三つのパターンがあります。各パターンでのオプションの仕様について次の三つの表に示します。

表 6-3　前回のプロセス監視定義ファイルからの変更の場合のオプションの仕様

オプション	オプションの仕様
なし	1. 警告メッセージを出力する。 2. プロセス管理テーブルは書き換えない。 修正された定義は，admreload コマンドを -f オプション付きで再度入力するか，または次回 ADM 正常開始もしくは強制正常開始するまで反映されない。
-f	1. プロセス管理テーブルを書き換える。 2. 連続異常終了回数のカウンタを初期化する。 3. 起動中のプロセスを，変更前の定義に従って停止させる（通常の停止処理と同じ方法で停止する）。 4. 変更後の定義に従ってプロセスを起動する（通常の起動処理と同じ方法で起動する）。

表 6-4　admstartprc コマンドによる動的監視プロセスと同じ識別子をプロセス監視定義ファイルに設定した場合のオプションの仕様

オプション	オプションの仕様
なし	1. 警告メッセージを出力する。 2. プロセス管理テーブルは書き換えない。 修正された定義は，admreload コマンドを -f オプション付きで再度入力するか，または次回 ADM 正常開始もしくは強制正常開始するまで反映されない。
-f	1. プロセス管理テーブルを書き換える。 2. 連続異常終了回数のカウンタを初期化する。 3. 起動中のプロセスを，システムコール（UNIX の場合は kill()，Windows の場合は TerminateProcess()）で停止させる。 4. 変更後の定義に従ってプロセスを起動する。

表 6-5　API（tpadm_start_monitor()）による動的監視プロセスと同じ識別子をプロセス監視定義ファイルに設定した場合

オプション	オプションの仕様
なし	1. 警告メッセージを出力する。 2. プロセス管理テーブルは書き換えない。 修正された定義は，admreload コマンドを -f オプション付きで再度入力するか，または次回 ADM 正常開始もしくは強制正常開始するまで反映されない。
-f	1. プロセス管理テーブルを書き換える。 2. 連続異常終了回数のカウンタを初期化する。 3. 起動中のプロセスは停止しない。監視対象から外される。 4. 定義変更後のプロセスは起動しない。

(3) 定義追加の動作について

プロセス管理テーブルを書き換えます。追加は即時有効となりますが，プロセス起動はしません。プロセス数が運用定義 /ADM/max_process_num で設定した数を超えた場合は，警告メッセージを出力し，追加は実行されません。この場合は，プロセス監視定義ファイルを再度編集して admreload コマンドを再度実行することによって，追加を有効にしてください。追加した定義にエラーがある場合は追加は実行されません。

6.4.2 再読み込み定義単位の指定

再読み込みを行う定義単位は，ADM 識別子ごと，およびプロセス監視定義ファイル全体の 2 種類があります。再読み込みの定義単位のオプションの仕様について次の表に示します。

表 6-6　再読み込みの定義単位のオプションの仕様

オプション	オプションの仕様
-i 識別子 ...	1. プロセス監視定義ファイル上の設定された識別子だけを再読み込みする。 2. 識別子は重複した識別子を含めないで 64 個まで指定できる。 3. エラーのある定義があっても，設定された識別子をすべて再読み込みして，プロセス管理テーブルに反映できる定義はすべて反映する。
-a	1. プロセス監視定義ファイルすべてを再読み込みする。 2. エラーのある定義があっても，全件を再読み込みして，プロセス管理テーブルに反映できる定義をすべて反映する。

6.4.3 状態遷移による動作

ADM の状態に対応した admreload コマンドの動作について表 6-7 に，監視対象プロセスの状態に対応した admreload コマンドの動作について表 6-8 に示します。

表 6-7　ADM の状態に対応した admreload コマンドの動作

ADM の状態	admreload コマンドの動作
未起動	KFCB29013-E Cannot access to ADMD. を表示して終了する。
起動～ admstart コマンド入力待ち（初期化中）	KFCB29013-E Cannot access to ADMD. を表示して終了する。
admstart コマンド入力待ち※	KFCB29001-I System has not invoked yet. を表示して終了する。
admstart コマンド入力～ admstop コマンド入力	コマンドを受け付ける。詳細は表 6-8 を参照のこと。
admstop コマンド入力～終了	KFCB29070-I ADMD has already terminated. または KFCB29005-E Communication error occurred. を表示して終了する。

注※
運用定義 /ADM/set_conf_mode が "AUTO" の場合，または運用定義 /ADM/set_conf_mode が "MANUAL" で前回異常終了の場合は，この状態は存在しません。

表 6-8　監視対象プロセスの状態に対応した admreload コマンドの動作

監視対象プロセスの状態	プロセス監視定義ファイルの変更状況	admreload コマンドのオプション	admreload コマンドの動作	admreload コマンド入力後のプロセスの状態
起動中	削除	なし	KFCB29065-W を表示し，プロセス管理テーブルには反映しない。	起動中
		-f	プロセス管理テーブルを更新する。	停止
	変更	なし	KFCB29065-W を表示し，プロセス管理テーブルには反映しない。	起動中
		-f	プロセス管理テーブルを更新する。	起動中
未起動	削除	なし	プロセス管理テーブルを更新する。	停止
		-f	プロセス管理テーブルを更新する。	停止
	変更	なし	プロセス管理テーブルを更新する。	停止
		-f	プロセス管理テーブルを更新する。	停止
	追加	なし	プロセス管理テーブルを更新する。	停止
		-f	プロセス管理テーブルを更新する。	停止

6.5　ADM の複数登録機能

複数の運用支援機能の実行環境を登録する機能について説明します。

6.5.1　ADM の複数登録機能の概要

ADM の複数登録機能は，TPBroker 実行環境の環境変数を使い分け，1 つの OS に運用支援機能の実行環境を複数登録する機能です。Windows 版の場合は，指定した名称で OS にサービスを登録します。なお，Windows 版については TPBroker 05-15 以降で有効です。

TPBroker 実行環境は，環境変数 TPSPOOL，TPFS，ADMSPOOL，および ADMFS の設定値により識別します。そのため，環境変数 TPSPOOL，TPFS，ADMSPOOL，および ADMFS は，tssetup コマンドおよび admsetup コマンドを実行する TPBroker 実行環境ごとに個別の値を設定する必要があります。

admsetup コマンドの実行時に設定された TPBroker 環境の環境変数は，運用支援機能の開始時に引き継がれます。運用支援機能の開始時に引き継ぐ環境変数については，「9.3 admsetup（実行環境のセットアップ）」を参照してください。

6.5.2　Windows 版固有の機能

TPBroker 05-13 までは，「TPBroker」の名称でサービスを登録していましたが，TPBroker 05-15 以降は，admsetup コマンドの実行時に -r オプションを指定してサービス名を指定できます。

また，admsetup に -r オプションを指定してサービスを登録した場合，ADM デーモンはイベントログのデータ部分にサービス名称を出力します。イベントログのデータ部分を参照することにより，どのサービスの ADM が出力したログであるかを識別できます (-r オプションを指定しない場合，データ部分には何も出力されません)。

6.5.3　設定手順

ここでは，2 つの運用支援機能の実行環境をセットアップし，TPBroker を開始・終了する手順を説明します。

表 6-9　表運用支援機能の実行環境で設定する環境変数の例

		運用支援機能の実行環境 1	運用支援機能の実行環境 2
UNIX 版の例	TPSPOOL	/opt/TPBrokerV5/otsspool	/opt/TPBrokerV5/otsspool2
	TPFS	/opt/TPBrokerV5/tpfs	/opt/TPBrokerV5/otsspool2
	ADMSPOOL	/opt/TPBrokerV5/spool	/opt/TPBrokerV5/spool2
	ADMFS	/opt/TPBrokerV5/admfs	/opt/TPBrokerV5/spool2
Windows 版の例	TPSPOOL	C:¥TPBrokerV5¥otsspool	C:¥TPBrokerV5¥otsspool2
	TPFS	C:¥TPBrokerV5¥tpfs	C:¥TPBrokerV5¥tpfs2
	ADMSPOOL	C:¥TPBrokerV5¥spool	C:¥TPBrokerV5¥spool2
	ADMFS	C:¥TPBrokerV5¥admfs	C:¥TPBrokerV5¥admfs2
	サービス名	TPBroker ADM Service1	TPBroker ADM Service2

注

環境変数 TPSPOOL，TPFS，ADMSPOOL，ADMFS には運用支援機能の実行環境ごとに個別の値を設定します

(1) UNIX 版の場合の手順

(a) 環境変数の設定

ターミナルを開き，「運用支援機能の実行環境 1」のための環境変数を設定します。以降，この環境を「運用支援機能の実行環境 1」と表記します。

別のターミナルを開き，運用支援機能の実行環境 2 のための環境変数を設定します。以降，この環境を「運用支援機能の実行環境 2」と表記します。

(b) セットアップ

「運用支援機能の実行環境 1」で，tssetup コマンドおよび admsetup コマンドを使用して，「運用支援機能の実行環境 1」の環境を初期化します。

例：

```
prompt> tssetup
prompt> admsetup -c  $TPDIR/adm/admconf1.cf
```

同様に，「運用支援機能の実行環境 2」で，tssetup コマンドおよび admsetup コマンドを使用して，「運用支援機能の実行環境 2」の環境を初期化します。

例：

```
prompt> tssetup
prompt> admsetup -c  $TPDIR/adm/admconf2.cf
```

(c) 運用

運用支援機能を手動で開始する場合，「運用支援機能の実行環境 1」で，admstart コマンドを実行し，運用支援機能を開始します。

同様に，「運用支援機能の実行環境 2」で運用支援機能を開始します。

運用支援機能を終了する場合は，運用支援機能の実行環境の設定を admstat コマンドや admlsprc コマンドで確認後，admstop コマンドを実行します。「運用支援機能の実行環境 1」で admstat コマンドを実行した場合，ADMSPOOL に "/opt/TPBrokerV5/spool" が設定されていることを確認できます。

例：

```
prompt> admstat
prompt> admlsprc -l
prompt> admstop
```

(d) アンセットアップ

運用支援機能を終了後，admsetup コマンドを使用して，運用支援機能の実行環境を削除します。

例：

```
prompt> admsetup -d
```

(2) Windows 版の場合の手順

(a) 環境変数の設定

コマンドプロンプトを開き，「運用支援機能の実行環境 1」の環境変数を設定します。以降，この環境を「運用支援機能の実行環境 1」と表記します。

別のコマンドプロンプトを開き，運用支援機能の実行環境 2 のための環境変数を設定します。以降，この環境を「運用支援機能の実行環境 2」と表記します。

(b) セットアップ

「運用支援機能の実行環境 1」で，tssetup コマンドおよび admsetup コマンドを使用して，TPBroker の環境を初期化します。admsetup は，-r オプションにサービス名 " TPBroker ADM Service1 " を指定します。

例：

```
prompt> tssetup
prompt> admsetup -c  "%TPDIR%¥adm¥admconf1.cf" -r "TPBroker ADM
Service1"
```

Windows のサービスとして，"TPBroker ADM Service1" が登録されます。

同様に，「運用支援機能の実行環境 2」で「運用支援機能の実行環境 2」の環境を初期化します。

例：

```
prompt> tssetup
prompt> admsetup -c  "%TPDIR%¥adm¥admconf2.cf" -r "TPBroker ADM
Service2"
```

(c)　運用

運用支援機能を手動で開始する場合は，net コマンドまたは，Windows の「サービス」ウィンドウを利用します。Windows 版の場合はサービス名を指定するため，環境変数の設定を確認する必要はありません。

例：

```
prompt> net start "TPBroker ADM Service1"
prompt> net start "TPBroker ADM Service2"
```

運用支援機能を終了する場合は，サービスを停止する前に停止するサービスに関連付けた運用支援機能の実行環境の設定を admlsenv コマンドや admlsprc で確認します。

例：

```
prompt> admlsenv -r "TPBroker ADM Service1"
prompt> admlsprc -l
prompt> net stop "TPBroker ADM Service1"
prompt> net stop "TPBroker ADM Service2"
```

(d)　アンセットアップ

運用支援機能を終了後，admsetup コマンドを使用して，運用支援機能の実行環境を削除します。admsetup コマンドは，-r オプションを使用して，サービス名を指定します。

例：

```
prompt> admsetup -d -r "TPBroker ADM Service1"
prompt> admsetup -d -r "TPBroker ADM Service2"
```

6.6　メッセージログの管理

運用支援機能では，メッセージログを管理しています。

TPBroker は，メッセージログをイベントログまたはログファイルに出力します。admlogcat コマンドを入力すると，メッセージログファイル中のメッセージを標準出力に出力します。

また，ADM デーモンプロセスでメッセージログファイルの生成に失敗すると，イベントログにメッセージを出力してデーモンプロセスが異常終了します。

運用支援機能はメッセージログファイルなどの世代管理をしています。世代管理については，「6.2.1(5)　作業ディレクトリの自動退避」を参照してください。

出力されるメッセージについては「11.　メッセージ」および「12.　Java OTS が出力するメッセージ」を参照してください。

6.7 稼働統計情報の取得

稼働統計情報として，システム情報および監視対象プロセス情報を取得できます。

6.7.1 システム情報の取得

TPBroker では，システムの情報を取得する機能を提供しています。admstat コマンドで，現在のシステムの情報を表示できます。admstat コマンドについては，「9.3　運用コマンドの詳細」を参照してください。

6.7.2 監視対象プロセス情報の取得

TPBroker では，監視中のプロセスに関する情報を取得する機能を提供しています。admlsprc コマンドで，現在監視中のプロセスに関する情報を表示できます。admlsprc コマンドについては，「9.3　運用コマンドの詳細」を参照してください。

6.8 UAP ログ出力機能（C++）（UNIX）

UAP ログ出力機能とは，ユーザアプリケーションプログラム（UAP）から，システムログおよび共通ログファイルにメッセージを出力する機能です。この機能には，三つの出力方式があります。ユーザは，三つの出力方式から任意に選択して利用できます。

UAP ログ出力機能は，Cosminexus TPBroker ではサポートしていません。

6.8.1 UAP ログの出力方式

UAP ログの出力方式には，システムログへの出力，共通ログファイルへの出力，および両者への一括出力があります。次のメッセージが出力されます。

KFCB5nnnn-X

（凡例）

5nnnn：50000 ～ 59999 の値

X：エラーレベル

(1) システムログへの出力

UNIX の syslog にメッセージを出力します。システムログへの出力に失敗した場合は，例外を返します。

(2) 共通ログファイルへの出力

指定したディレクトリに二つの共通ログファイルを作成し，そのファイルへメッセージを出力します。共通ログファイルの名称は，「tpadm1.log」および「tpadm2.log」です。共通ログファイルを作成するディレクトリは，任意に指定できます。また，異なるディレクトリを指定することで，複数の共通ログファイルを使用できます。

共通ログファイルのサイズは，デフォルトで 1 ファイル当たり 512 キロバイトです。1 ファイル当たりのサイズは変更できます。

共通ログファイルは通常のテキストファイルであるため，システムダウン時，および OS ダウン時には内容は保証されません。また，ファイルへの出力に失敗した場合は，例外を返します。

共通ログファイルはテキストファイルであるため，内容を編集するためのコマンドは提供しません。

(3) システムログおよび共通ログファイルへの一括出力

アプリケーションプログラムからの一度のログ出力で，同じ内容をシステムログと共通ログファイルに出力します。(1)，(2) で説明した両方の特徴を持ちます。システムログ，共通ログの順に出力します。システムログへの出力に失敗した場合でも，共通ログへの出力は行います。どちらかの出力処理でエラーが発生した場合は，例外を返します。

例外の情報によって，どちらの出力に失敗したかを判断できます。

6.8.2 マルチスレッドおよびマルチプロセス環境への対応

UAP ログ出力機能は，マルチスレッドおよびマルチプロセス環境に対応しています。このため，UAP ログ出力機能を使用する場合に，排他制御を行う必要はありません。

6.8.3 出力形式

システムログへの出力形式，および共通ログファイルへの出力形式を次に示します。

(1) システムログ

システムログへの出力形式は，次のようになります。

形式

```
AAA BBB CCC[DDD]: EEE FFF GGG:HHH
```

意味

AAA：日時

BBB：ホスト名

CCC：プログラム名

DDD：プロセス ID

EEE：メッセージ ID

FFF：プロセス ID

GGG：スレッド ID

HHH：メッセージ本文

出力例

```
Apr 26 13:30:00 myhost1 TpLog[1111]: KFCB50001-I 1111 3:AP start.
Apr 26 13:30:02 myhost1 TpLog[1111]: KFCB50010-I 1111 3:AP error.
Apr 26 13:30:03 myhost1 TpLog[1111]: KFCB50002-I 1111 3:AP stop.
```

(2) 共通ログファイル

共通ログファイルへの出力形式は，次のようになります。

形式

```
aaa bbb ccc ddd : eee
```

意味

aaa：日時

bbb：メッセージ ID

ccc：プロセス ID

ddd：スレッド ID

eee：メッセージ本文

出力例

```
Apr 26 13:30:00 1999 KFCB50001-I 1111 3:AP start.
Apr 26 13:30:02 1999 KFCB50010-I 1111 3:AP error.
Apr 26 13:30:03 1999 KFCB50002-I 1111 3:AP stop.
```

7 TPBroker の運用

この章では，TPBroker の運用の流れ，TPBroker およびアプリケーションプログラムの開始と終了，および TPBroker の運用方法について説明します。

7.1 TPBroker の運用の流れ

ここでは，TPBroker インストール後の運用の流れについて，TPBroker の開始から終了までを中心に説明します。次の三つの運用パターンごとに説明します。

- ORB および ADM を使用して TPBroker を運用する場合
- ORB，ADM および OTS を使用して TPBroker を運用する場合
- ORB および OTS を使用して TPBroker を運用する場合

TPBroker の運用支援機能（ADM）を使用した運用支援機能実行環境について

運用支援機能実行環境では，システム運用時のオペレータの操作を省力化し，システムの運用コストを削減できる機能を提供しています。運用支援機能実行環境を OS に登録されることをお勧めします。

なお，この節に記載されているコマンドの詳細については，「9.3　運用コマンドの詳細」を参照してください。

7.1.1 ORB および ADM を使用して TPBroker を運用する場合

(1) TPBroker の環境設定

TPBroker を開始する前に，次に示す手順で TPBroker の環境設定をします。

1. **環境変数の設定**
   設定する環境変数の詳細については，「2.2　環境変数を設定する」を参照してください。
2. **tssetup コマンドの実行**
   tssetup コマンドで，TPBroker 環境を初期化します。

(2) システム環境定義の変更（任意）

システム環境定義を変更します。

システム環境定義は TPBroker 開始前に変更できます。TPBroker 稼働中に tsdefvalue コマンドで変更することもできます。システム環境定義の詳細については，「8.3　システム環境定義の詳細」を参照してください。

(3) プロセス監視定義の設定

プロセス監視定義を設定します。詳細は，「8.2　プロセス監視定義の詳細」を参照してください。

(4) 運用支援機能実行環境の OS への登録

TPBroker の運用支援機能実行環境を OS に登録するには，admsetup コマンドを使用します。

(5) TPBroker の開始

自動開始または手動開始によって ORB 環境と OTS 環境（トランザクションサービス）が同時に起動され，プロセス監視が開始されます。

自動開始と手動開始について
　TPBroker の開始は，運用定義で自動開始または手動開始のどちらかを設定できます。手動開始の場合，admstart コマンドを使用して TPBroker を開始します。運用定義の詳細については「8.3.1　運用定義」を参照してください。

(6) TPBroker の開始の確認

TPBroker の開始を，次のコマンドで確認します。

- admlogcat
  プロセスが起動されたことを確認します。
- admlsprc
  プロセスが監視中であることを確認します。

(7) アプリケーションプログラムの開始

TPBroker が開始されたあとに，アプリケーションプログラムを実行します。

(8) アプリケーションプログラムの終了

アプリケーションプログラムの終了方法は，VisiBroker のアプリケーションプログラムの終了方法に従います。VisiBroker のアプリケーションプログラムの終了方法については，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

(9) TPBroker の終了

admstop コマンドを使用し，TPBroker を終了します。

(10) TPBroker の終了の確認

TPBroker の終了を，次のコマンドで確認します。

- admlogcat
  プロセスが停止されたことを確認します。

7.1.2 ORB，ADM および OTS を使用して TPBroker を運用する場合

(1) TPBroker の環境設定

TPBroker を開始する前に，次に示す手順で TPBroker の環境設定をします。

1. **環境変数の設定**
   設定する環境変数の詳細については，「2.2　環境変数を設定する」を参照してください。
2. **tssetup コマンドの実行**
   tssetup コマンドで，TPBroker 環境を初期化します。

(2) システム環境定義の変更（任意）

システム環境定義を変更します。

システム環境定義は TPBroker 開始前に変更できます。TPBroker 稼働中に tsdefvalue コマンドで変更することもできます。システム環境定義の詳細については，「8.3　システム環境定義の詳細」を参照してください。

(3) リソースマネージャ連携の準備（C++）

リソースマネージャと連携するには，次の作業をする必要があります。

1. **tslnkrm コマンドを使用して，リソースマネージャを TPBroker に登録する。**
2. **リソースマネージャをシステム環境定義に登録する。**
3. **オブジェクトファイルを作成する。**
   オブジェクトファイルには，次の三つがあります。
   - tslnkrm コマンド実行によって作成された標準トランザクション制御用オブジェクトファイル
   - tsmkobj コマンドで作成したオブジェクトファイル
   - 各リソースマネージャが提供するオブジェクトファイル
4. **オブジェクトファイルとアプリケーションプログラムをリンクする。**
   tslnkrm コマンド実行によって作成された標準トランザクション制御用オブジェクトファイルと tsmkobj コマンドで作成したオブジェクトファイルついては，どちらか一つをアプリケーションプログラムとリンクさせます。

(4) プロセス監視定義の設定

プロセス監視定義を設定します。詳細は，「8.2　プロセス監視定義の詳細」を参照してください。

(5) 運用支援機能実行環境の OS への登録

TPBroker の運用支援機能実行環境を OS に登録するには，admsetup コマンドを使用します。

(6) TPBroker の開始

自動開始または手動開始によって ORB 環境と OTS 環境（トランザクションサービス）が同時に起動され，プロセス監視が開始されます。

自動開始と手動開始について

TPBroker の開始は，運用定義で自動開始または手動開始のどちらかを設定できます。手動開始の場合，admstart コマンドを使用して TPBroker を開始します。運用定義の詳細については「8.3.1　運用定義」を参照してください。

(7) TPBroker の開始の確認

TPBroker の開始を，次のコマンドで確認します。

- admlogcat
  プロセスが起動されたことを確認します。
- admlsprc
  プロセスが監視中であることを確認します。

(8) アプリケーションプログラムの開始

TPBroker が開始されたあとに，アプリケーションプログラムを実行します。

(9) アプリケーションプログラムの終了

アプリケーションプログラムの終了方法は，VisiBroker のアプリケーションプログラムの終了方法に従います。VisiBroker のアプリケーションプログラムの終了方法については，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

(10) TPBroker の終了

admstop コマンドを使用し，TPBroker を終了します。

(11) TPBroker の終了の確認

TPBroker の終了を，次のコマンドで確認します。

- admlogcat
  プロセスが停止されたことを確認します。

7.1.3 ORB および OTS を使用して TPBroker を運用する場合

(1) TPBroker の環境設定

TPBroker を開始する前に，次に示す手順で TPBroker の環境設定をします。

1. **環境変数の設定**
   設定する環境変数の詳細については，「2.2　環境変数を設定する」を参照してください。
2. **tssetup コマンドの実行**
   tssetup コマンドで，TPBroker 環境を初期化します。

(2) システム環境定義の変更（任意）

システム環境定義を変更します。

システム環境定義は TPBroker 開始前に変更できます。TPBroker 稼働中に tsdefvalue コマンドで変更することもできます。システム環境定義の詳細については，「8.3　システム環境定義の詳細」を参照してください。

(3) リソースマネージャ連携の準備（C++）

リソースマネージャと連携するには，次の作業をする必要があります。

1. **tslnkrm コマンドを使用して，リソースマネージャを TPBroker に登録する。**
2. **リソースマネージャをシステム環境定義に登録する。**
3. **オブジェクトファイルを作成する。**
   オブジェクトファイルには，次の三つがあります。
   - tslnkrm コマンド実行によって作成された標準トランザクション制御用オブジェクトファイル
   - tsmkobj コマンドで作成したオブジェクトファイル
   - 各リソースマネージャが提供するオブジェクトファイル
4. **オブジェクトファイルとアプリケーションプログラムをリンクする。**
   tslnkrm コマンド実行によって作成された標準トランザクション制御用オブジェクトファイルと tsmkobj コマンドで作成したオブジェクトファイルついては，どちらか一つをアプリケーションプログラムとリンクさせます。

(4) TPBroker の開始

OSAgent を起動することで ORB 環境を起動してから，tsstart コマンドを使用して OTS 環境（トランザクションサービス）を起動します。OSAgent の詳細は，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

(5) TPBroker の開始の確認

TPBroker の開始を，次のコマンドで確認します。

● tslogcat
　プロセスが起動されたことを確認します。

● tslsrm
　リソースマネージャの情報を表示します。

(6) アプリケーションプログラムの開始

TPBroker が開始されたあとに，アプリケーションプログラムを実行します。

(7) アプリケーションプログラムの終了

アプリケーションプログラムの終了方法は，VisiBroker のアプリケーションプログラムの終了方法に従います。VisiBroker のアプリケーションプログラムの終了方法については，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

(8) TPBroker の終了

tsstop コマンドを使用し，OTS 環境（トランザクションサービス）を終了します。

(9) TPBroker の終了の確認

TPBroker の終了を，次のコマンドで確認します。

● tslogcat
　プロセスが停止されたことを確認します。

7.2　TPBroker の開始と終了

ここでは，TPBroker の開始方法および終了方法について説明します。

7.2.1　TPBroker の環境の開始と終了

TPBroker は OTS と ORB の環境を開始してから運用します。環境の開始方法には次に示す二つがあります。

1. 運用支援機能を使用して開始する方法
2. OTS と ORB を別々に起動して開始する方法

この節では，1. の方法について説明します。2. の方法については，「7.5　トランザクションサービスの運用」を参照してください。

また，以降の説明に記載されているコマンドの詳細については，「9.3　運用コマンドの詳細」を参照してください。

7.2.2　TPBroker の開始

運用支援機能を使用した TPBroker の開始方法，開始モード，および開始形態の決定について説明します。ここでは，OTS と ORB が，運用支援機能で使用するプロセス監視定義ファイルに設定されているものとします。

なお，この項で説明する admstart コマンドについては，「9.3　運用コマンドの詳細」を，運用定義については「8.3.1　運用定義」を参照してください。

(1)　開始方法

TPBroker の開始方法は，運用定義 /ADM/set_conf_mode の設定によって，自動開始と手動開始の二つに分けられます。

- 自動開始
  運用定義 /ADM/set_conf_mode で "AUTO" を設定した場合，OS 起動時に自動的に開始します。
- 手動開始
  運用定義 /ADM/set_conf_mode で "MANUAL"，"MANUAL2"，"MANUAL3" または "MANUAL4" を設定した場合，開始コマンド（admstart）を入力して開始します。
  手動開始の場合，次のような指定ができます。

  正常開始または再開始の場合
  　　admstart

  強制正常開始の場合
  　　admstart -f

(2) TPBroker の開始モード

TPBroker の開始モードには，次の二つがあります。

- 正常開始
  前回のオンラインが正常に終了して引き継ぐ情報がないとき，または新たに TPBroker を開始するときのモードです。
- 全面回復による再開始
  前回のオンラインの終了状態を引き継いで開始するときのモードです。

(3) 開始形態の決定

開始形態は，前回の終了モードや運用定義の設定などによって異なります。開始形態の決定について，次の表に示します。

表 7-1 開始形態の決定

前回の終了形態	運用定義の設定	今回の開始形態	開始モード
admstop	AUTO	自動	正常開始
	MANUAL	admstart	正常開始
		admstart -f	正常開始
	MANUAL2	admstart	正常開始
		admstart -f	正常開始
	MANUAL3 ※5	admstart	正常開始
		admstart -f	正常開始
	MANUAL4 ※5	admstart	正常開始
		admstart -f	正常開始
admstop -f	AUTO	自動	再開始※1
	MANUAL	admstart	再開始※1
		admstart -f	強制正常開始
	MANUAL2	admstart	再開始※1
		admstart -f	強制正常開始
	MANUAL3 ※5	admstart	再開始※1
		admstart -f	強制正常開始
	MANUAL4 ※5	admstart	再開始※1
		admstart -f	強制正常開始
admstop -fr	AUTO	自動	再開始※2
	MANUAL	admstart	再開始※2
		admstart -f	強制正常開始

前回の終了形態	運用定義の設定	今回の開始形態	開始モード
	MANUAL2	admstart	再開始※ 2
		admstart -f	強制正常開始
	MANUAL3 ※ 5	admstart	再開始※ 2
		admstart -f	強制正常開始
	MANUAL4 ※ 5	admstart	再開始※ 2
		admstart -f	強制正常開始
異常終了※ 3	AUTO	自動	再開始※ 2
	MANUAL	自動	再開始※ 2
	MANUAL2	admstart	再開始※ 2
		admstart -f	強制正常開始
	MANUAL3 ※ 4 ※ 5 (TPBroker 単独ダウン)	自動	再開始※ 2
	MANUAL3 ※ 4 ※ 5(TPBroker の稼動中に OS シャットダウン)	自動	再開始※ 2
	MANUAL3 ※ 4 ※ 5 (TPBroker の稼動中に OS 異常終了)	admstart	再開始※ 2
		admstart -f	強制正常開始
	MANUAL4 ※ 5 (TPBroker 単独ダウン)	自動	再開始※ 2
	MANUAL4 ※ 3 ※ 5 (TPBroker の稼動中に OS シャットダウン)	admstart	再開始※ 2
		admstart -f	強制正常開始
	MANUAL4 ※ 5 (TPBroker の稼動中に OS 異常終了)	admstart	再開始※ 2
		admstart -f	強制正常開始

注※ 1

プロセス監視定義ファイルに設定されている再開始用のプロセスを起動または監視します。

注※ 2

UNIX では前回終了時の運用支援機能のプロセス監視情報に設定されているプロセスを起動または監視します。Windows 版では OS シャットダウン時には TPBroker

が正常に終了されるため，正常開始します。

注※ 3

TPBroker が開始している状態で，OS をシャットダウンすると，前回の終了形態はUNIX 版では異常終了，Windows 版では正常終了とみなします。

注※ 4

Windows 版では開始モード MANAUAL3 をサポートしていません。UNIX 版のみでの動作となります。

注※ 5

MANUAL3 および MANUAL4 は，TPBroker 05-15 以降でサポートします。

(4) 注意事項

OS を再起動する場合は「shutdown -r」コマンドを使用してください。ADM デーモンは「reboot」コマンドを使って OS の再起動を行った場合，次回の再起動は正しく動作しません。以下の現象が発生することがあります。

- 開始モードに「MANUAL3」を設定した場合，「MANUAL4」と同様の開始形態となる場合があります。
- reboot コマンドを実行する前に監視していたプロセスがブート後に監視されない場合があります。

7.2.3 TPBroker の終了

運用支援機能を使用した TPBroker の終了方法および終了モードについて説明します。

(1) 終了方法

TPBroker の終了は，admstop コマンドで行います。このとき，次の二つの終了モードのどちらかを指定できます。

- 正常終了
  admstop と入力します。
- 強制正常終了
  admstop -f または admstop -fr と入力します。

admstop コマンドの指定方法については，「9.3　運用コマンドの詳細」を参照してください。オプションの指定がない場合は，正常終了になります。

Windows 版の場合，サービス「TPBroker」を停止すると，TPBroker は強制正常終了します。このとき，admstop コマンドが自動的に発行され，ADM デーモンが停止します。TPBroker を正常終了する場合は，admstop コマンドを実行してからサービス「TPBroker」を停止してください。サービス「TPBroker」を停止した場合の TPBroker の終了モードは，運用定義 /ADM/service_stop_mode の設定に依存します。

（2） TPBroker の終了モード

TPBroker の終了モードには，次の三つがあります。

- 正常終了
  admstop と入力した場合の終了モードです。
  プロセス監視定義ファイルに正常終了用コマンドが定義されている場合は，運用支援機能はそのコマンドを発行します。
- 強制正常終了
  admstop -f または admstop -fr と入力した場合の終了モードです。
  プロセス監視定義ファイルに強制正常終了用コマンドが定義されている場合は，運用支援機能はそのコマンドを発行します。
- 異常終了
  admstop と入力しないで，TPBroker が終了した場合の終了モードです。

（3） 注意事項

- TPBroker の停止コマンドを使用しないで，UNIX の kill コマンドなどで直接プロセスを停止させた場合には，TPBroker はダウンします。
- Windows 版の場合，サービス「TPBroker」の停止に 1 時間以上掛かると，ADM デーモンを強制停止します。この場合，監視中のプロセスは停止しません。また，次回起動時の開始モードは再開始になります。
- Windows 版の場合，Windows の GUI を持つアプリケーションのプロセスを監視対象にしているときは，そのアプリケーションを監視対象から外すか，またはサービス「TPBroker」を停止してから，ログオフしてください。サービス「TPBroker」に対する Windows のサービスの設定で「デスクトップとの対話をサービスに許可」チェックボックスをオンにしている状態で，TPBroker を稼働したままログオフすると，監視対象プロセスが Windows の GUI を持つアプリケーションだった場合，このアプリケーションは異常終了することがあります。
- Windows 版の場合，マシンをシャットダウンすると，TPBroker のデーモンプロセスや監視対象プロセスも OS によって強制停止されます。サービス「TPBroker」を停止してから，マシンをシャットダウンしてください。

7.3　アプリケーションプログラムの開始と終了

ここでは，OTS のアプリケーションプログラムの開始と終了について説明します。ORB については，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

アプリケーションプログラムの開始と終了に関する運用は，Cosminexus TPBroker ではサポートしていません。

7.3.1　アプリケーションプログラムの開始

トランザクションサービスが開始されたあとに，アプリケーションプログラムを実行してください。このとき，環境変数 TPDIR，TPSPOOL が適切なディレクトリに設定されている必要があります。環境変数 TPSPOOL が設定されていない場合は，$TPDIR/otsspool が使用されます。環境変数の詳細については，「2.2　環境変数を設定する」および「9.3　運用コマンドの詳細」の tssetup コマンドを参照してください。

Java アプリケーションを実行する場合は，vbj コマンド行で次に示すオプションを設定してください。

```
org.omg.PortableInterceptor.ORBInitializerClass.COM.Hitachi.softwa
re.TPBroker.OTS.Init
```

例

```
vbj
-Dorg.omg.PortableInterceptor.ORBInitializerClass.COM.Hitachi.s
oftware.TPBroker.OTS.Init Client
```

アプリケーションプログラムの運用に関する注意

OTS の機能を使用した場合，トランザクション終了前にリカバラブルサーバがダウンすると，ダウンするタイミングによって syslog またはイベントログにメッセージ KFCB31215-E が繰り返し出力され，トランザクションブランチが残る場合があります。この場合はダウンしたリカバラブルサーバを再開始してください。

7.3.2　アプリケーションプログラムの終了

VisiBroker のアプリケーションプログラムの終了方法に従います。VisiBroker のアプリケーションプログラムの終了方法については，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

7.4 リソースマネージャの運用（C++）

ここでは，XA インタフェースによる連携やリソースマネージャの登録・削除など，リソースマネージャの運用について説明します。

この機能は C++ 実行環境で使用できます。

リソースマネージャと連携する場合の注意事項については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

リソースマネージャとの連携は，Cosminexus TPBroker ではサポートしていません。

7.4.1 XA インタフェースをサポートしたリソースマネージャの場合

XA インタフェースをサポートしたリソースマネージャの場合は，TPBroker のトランザクションと同期を取ってアップデートできます。同期を取る場合は，OTS の Current または Terminator インタフェースを使います。リソースマネージャが提供する同期点制御の機能は，OTS の Current または Terminator インタフェースと併用できません。リソースマネージャが提供する同期点制御の機能を使用した場合，リソースの不整合が起きる場合があります。

TPBroker のトランザクション処理で制御できるリソースマネージャは，XA インタフェースをサポートした製品（Oracle など）に限ります。

リソースマネージャが XA インタフェースをサポートしている場合，複数のリソースマネージャへアクセスするアプリケーションプログラムでは，それらの整合性を保ちながらアップデートできます。障害が原因でアプリケーションプログラムが異常終了した場合や，トランザクションサービスを再開始した場合でも，リソースマネージャのトランザクションを TPBroker で決着します。

7.4.2 XA インタフェースをサポートしていない，または XA インタフェースで TPBroker と連携していないリソースマネージャの場合

XA インタフェースをサポートしていないリソースマネージャの場合，リソースマネージャへのアクセスはできますが，TPBroker のトランザクションとは同期を取れません。XA インタフェースをサポートしていても，それを使って連携していないリソースマネージャの場合も同様です。

XA インタフェースで連携していないため，リソースマネージャへのアクセス中に障害が原因でアプリケーションプログラムが異常終了した場合，およびトランザクションサービスを再開始した場合には，TPBroker からリソースマネージャへトランザクションの決

着を指示しません。そのため，リソースマネージャ独自の機能でトランザクションを回復する必要があります。

7.4.3　XA インタフェースによって TPBroker と連携して使う場合の準備

XA インタフェースをサポートしたリソースマネージャを XA インタフェースで TPBroker と連携して使う場合に，準備する項目を次に示します。

(1)　TPBroker への登録

リソースマネージャを TPBroker と連携して使う場合，そのリソースマネージャを TPBroker に登録しなければなりません。リソースマネージャを登録するためには，tslnkrm コマンドを入力します。tslnkrm コマンドの詳細については，「9.3　運用コマンドの詳細」を参照してください。また，tslnkrm コマンドのオプションに指定する値については，リソースマネージャのマニュアルを参照してください。

(2)　アプリケーションプログラムのリンク

アプリケーションプログラムの実行形式ファイルを作成するときに，トランザクション制御用オブジェクトファイルおよびリソースマネージャのライブラリとオブジェクトモジュールをリンクする必要があります。

トランザクション制御用オブジェクトファイルは，tsmkobj コマンドを入力して作成します。または tslnkrm コマンドが作成する標準トランザクション制御用オブジェクトファイルを使用してもかまいません。tsmkobj，tslnkrm コマンドの詳細については，「9.3　運用コマンドの詳細」を参照してください。

また，リソースマネージャのライブラリおよびオブジェクトモジュールについては，使用するリソースマネージャのマニュアルを参照してください。

(3)　システム環境定義への登録

リソースマネージャを使う場合，システム環境定義のリソースマネージャ定義にそのリソースマネージャに関する情報を登録する必要があります。登録する内容には，リソースマネージャ固有の項目もあります。このような項目は，使用するリソースマネージャのマニュアルを参照してください。

リソースマネージャ定義については，「8.3.3　リソースマネージャ定義（C++)」を参照してください。

7.4.4　リソースマネージャの操作

(1)　リソースマネージャの情報の表示

TPBroker に登録されている，またはアプリケーションプログラムにリンクされているリ

ソースマネージャの情報を確認するには，tslsrm コマンドを使用します。このコマンドを実行すると，リソースマネージャ名，スイッチ名，オブジェクト名といったリソースマネージャの情報が標準出力に出力されます。

(2) リソースマネージャの登録と削除

TPBroker をインストールおよびセットアップしたとき，回復プロセスおよび決着プロセスは，TPBroker が提供する OTSCRM の XA インタフェース用のオブジェクトファイルをリンクしています。

TPBroker でそのほかのリソースマネージャと連携したトランザクションを実行する場合は，トランザクションサービスを開始する前に，TPBroker の tslnkrm コマンドを使用して実行環境にリソースマネージャを登録してください。TPBroker では，最大 7 個のリソースマネージャを登録できます。

TPBroker 下で実行されるトランザクションからアクセスしなくなったリソースマネージャを削除する場合，tslnkrm コマンドを -d オプション付きで実行します。

(3) トランザクション制御用オブジェクトファイルの作成

トランザクションサービスを使用するアプリケーションプログラムが，トランザクション内でリソースマネージャにアクセスする場合，そのアプリケーションプログラムにトランザクション制御用オブジェクトファイルをリンクする必要があります。ただし，トランザクション制御用共用ライブラリを使用する場合，アプリケーションプログラムに再びリンクする必要はありません。トランザクション制御用共用ライブラリの詳細については，「(4) トランザクション制御用共用ライブラリ」を参照してください。

TPBroker に登録されているすべてのリソースマネージャにアクセスするアプリケーションプログラムの場合，TPBroker が提供する標準トランザクション制御用オブジェクトファイル（$TPSPOOL/XA/tp_allrm.o）をリンクします。このオブジェクトファイルは tslnkrm コマンドを実行するたびに新しくアップデートされるので，リソースマネージャの登録または削除をした場合には，アプリケーションプログラムを再びリンクする必要があります。

TPBroker に登録されているリソースマネージャのうち，一部のリソースマネージャだけにアクセスするアプリケーションプログラムの場合，tsmkobj コマンドで新たなトランザクション制御用オブジェクトを作成して，それをアプリケーションプログラムに再びリンクします。

TPBroker に新たにリソースマネージャを登録した場合，およびリソースマネージャを削除した場合，そのリソースマネージャを含んだトランザクション制御用オブジェクトファイルをリンクしているアプリケーションプログラムを再びリンクする必要があります。

トランザクション内でリソースマネージャにアクセスしないアプリケーションプログラムの場合，TPBroker が提供しているオブジェクトファイル（$TPSPOOL/XA/tp_crm.o）

をリンクする必要があります。このオブジェクトファイルをリンクしない場合，トランザクションサービスの機能は使用できません。

(4) トランザクション制御用共用ライブラリ

トランザクション制御用オブジェクトファイルの代わりに，トランザクション制御用共用ライブラリを使用できます。トランザクション制御用共用ライブラリを使用すると，リソースマネージャの登録または削除をした場合でも，アプリケーションプログラムを再びリンクする必要がありません。ただし，リソースマネージャを削除した場合は，トランザクション制御用共用ライブラリを再作成する必要があります。

トランザクション制御用共用ライブラリは，tslnkrm または tsmkobj コマンドを実行したときに作成されます。作成されるディレクトリは $TPSPOOL/XA（Windows の場合，%TPSPOOL%¥XA および %TPSPOOL%¥bin）です。標準トランザクション制御用共用ライブラリは，「libtpallrm_r. 拡張子（拡張子は OS によって異なります）」という名称で作成されます。この共用ライブラリはマルチスレッドのアプリケーションプログラム用です。

トランザクション制御用共用ライブラリを使用する場合は，アプリケーションプログラムに $TPSPOOL/XA/tp_xaobj.o をリンクする必要があります。このオブジェクトをリンクしていない場合，アプリケーションプログラムでの OTS の初期化時にメッセージ KFCB31219-E が出力され，アプリケーションプログラムが異常終了します。

トランザクション制御用共用ライブラリとトランザクション制御用オブジェクトファイルを同時にアプリケーションプログラムにリンクすることはできません。必ずどちらか一方を選択してください。

7.4.5 XA トレース

XA トレースとは，XA インタフェースで連携しているときに TPBroker とリソースマネージャとの間で発行された XA 関数のトレースのことです。OTS は，リソースマネージャに対する XA 関数の呼び出し結果がエラーの場合に，トレースを出力します。XA トレースは，$TPSPOOL/log の下に XAtrace という名称でテキストファイルとして出力されます。XA トレースファイルは，最大 1 メガバイトです。1 メガバイトを超えると使用中のファイルを XAtrace.bak にコピーし，新規に XA トレースファイルを作成します。

XA トレースの出力形式の例を次に示します。関数ごとに次のように出力されます。

(1) xa_open ／ xa_close の場合

出力形式

```
YYYY/MM/DD hh:mm:ss.SSS aaa(bbb) ccccccc rmid=dd
flags=0xeeeeeeee rtn=ff
RMName=gggggg xa_info="hhhhhh...hhhhhh"
```

（凡例）

YYYY/MM/DD：日付

hh:mm:ss.SSS：時刻（ミリ秒）

aaa：プロセス ID

bbb：スレッド ID

ccccccc：関数名

dd：リソースマネージャ識別子

0xeeeeeeee：オプションフラグの値

ff：リターン値

gggggg：リソースマネージャ名（定義名）

hhhhhh...hhhhhh：オープン（クローズ）文字列

(2) xa_wait ／ xa_wait_recovery ／ ax_unreg の場合

出力形式

```
YYYY/MM/DD hh:mm:ss.SSS aaa(bbb) ccccccc rmid=dd
flags=0xeeeeeeee rtn=ff
```

（凡例）

YYYY/MM/DD：日付

hh:mm:ss.SSS：時刻（ミリ秒）

aaa：プロセス ID

bbb：スレッド ID

ccccccc：関数名

dd：リソースマネージャ識別子

0xeeeeeeee：オプションフラグの値

ff：リターン値

(3) xa_recover の場合

出力形式

```
YYYY/MM/DD hh:mm:ss.SSS aaa(bbb) ccccccc rmid=dd
flags=0xeeeeeeee rtn=ff
RMName=gggggg count=hh
XID=(0xmmmmmmmm,0xnnnnnn.......oooooo,0xpppppp......qqqqqq)
```

（凡例）

YYYY/MM/DD：日付

hh:mm:ss.SSS：時刻（ミリ秒）

aaa：プロセス ID

bbb：スレッド ID

ccccccc：関数名

dd：リソースマネージャ識別子

0xeeeeeeee：オプションフラグの値

ff：リターン値

gggggg：リソースマネージャ名（定義名）

hh：XID 数

0xmmmmmmmm：XID のフォーマット ID

0xnnnnnn.......oooooo：トランザクショングローバル識別子

0xpppppp......qqqqqq：トランザクションブランチ識別子

(4) xa_complete の場合

出力形式

```
YYYY/MM/DD hh:mm:ss.SSS aaa(bbb) ccccccc rmid=dd
flags=0xeeeeeeee rtn=ff
RMName=gggggg handle=hh retval=ii
```

（凡例）

YYYY/MM/DD：日付

hh:mm:ss.SSS：時刻（ミリ秒）

aaa：プロセス ID

bbb：スレッド ID

ccccccc：関数名

dd：リソースマネージャ識別子

0xeeeeeeee：オプションフラグの値

ff：リターン値

gggggg：リソースマネージャ名（定義名）

hh：非同期オペレーションのハンドル値

ii：非同期オペレーションのリターン値

(5) xa_start_2 ／ ax_reg_2 の場合

出力形式

```
YYYY/MM/DD hh:mm:ss.SSS aaa(bbb) ccccccc rmid=dd
flags=0xeeeeeeee rtn=ff
RMName=gggggg XID=
   (0xhhhhhhhh, 0xiiiiii......jjjjjj, 0xkkkkkk......llllll)
XACTL=(0xmmmmmmmm,nn)
```

（凡例）

YYYY/MM/DD：日付

hh:mm:ss.SSS：時刻（ミリ秒）

aaa：プロセス ID

bbb：スレッド ID

ccccccc：関数名

dd：リソースマネージャ識別子

0xeeeeeeee：オプションフラグの値

ff：リターン値

gggggg：リソースマネージャ名（定義名）

0xhhhhhhhh：XID のフォーマット ID

0xiiiiii.......jjjjjj：トランザクショングローバル識別子

0xkkkkkk......llllll：トランザクションブランチ識別子

0xmmmmmmmm：オプションフラグの値

nn：タイムアウト値

(6) 上記以外の場合

出力形式

```
YYYY/MM/DD hh:mm:ss.SSS aaa(bbb) ccccccc rmid=dd
flags=0xeeeeeeee rtn=ff
RMName=gggggg XID=
   (0xhhhhhhhh, 0xiiiiii.......jjjjjj, 0xkkkkkk......llllll)
```

（凡例）

YYYY/MM/DD：日付

hh:mm:ss.SSS：時刻（ミリ秒）

aaa：プロセス ID

bbb：スレッド ID

ccccccc：関数名

dd：リソースマネージャ識別子

0xeeeeeeee：オプションフラグの値

ff：リターン値

gggggg：リソースマネージャ名（定義名）

0xhhhhhhhh：XID のフォーマット ID

0xiiiiii.......jjjjjj：トランザクショングローバル識別子

0xkkkkkk......llllll：トランザクションブランチ識別子

7.5 トランザクションサービスの運用

ここでは，トランザクションサービスの開始・終了やトランザクションの決着などトランザクションサービスの運用について説明します。

7.5.1 トランザクションサービスの開始と終了

TPBroker は，トランザクションサービスを開始してトランザクションを制御します。トランザクションサービスの開始は，OSAgent と tsstart コマンドを使用します。tsstart コマンド入力前には OSAgent を起動しておく必要があります。

トランザクションサービスを終了するときには，tsstop コマンドを使用します。

tsstart，tsstop コマンドの詳細は，「9.3　運用コマンドの詳細」を参照してください。また，OSAgent の詳細は，マニュアル「Borland(R) Enterprise Server VisiBroker(R) デベロッパーズガイド」を参照してください。

7.5.2 トランザクションの状態表示

TPBroker が管理しているトランザクションに関する情報およびトランザクションブランチ数を tslstrn コマンドで表示できます。tslstrn コマンドの詳細は，「9.3　運用コマンドの詳細」を参照してください。

7.5.3 トランザクションの決着

tslstrn コマンドで参照したトランザクションの状態が PREPARED の場合，TPBroker はコマンドを使用してトランザクションを決着できます。コミットに決着させるときは tscommit コマンド，ロールバックに決着させるときは tsrollback コマンドを使用してください。tscommit，tsrollback コマンドの詳細は，「9.3　運用コマンドの詳細」を参照してください。

7.6 TPBroker ファイルシステム（UNIX）

ここでは，TPBroker ファイルシステムの概要や作成方法について説明します。

なお，この機能は UNIX 版でだけ有効です。

7.6.1 TPBroker ファイルシステムの概要

TPBroker は，OS が提供する使用効率の良いファイルシステム，および TPBroker が提供する信頼性の高いファイルシステムを使用しています。TPBroker で管理するファイルシステムを TPBroker ファイルシステムといいます。

(1) TPBroker ファイルシステム

TPBroker ファイルシステムは，OS が提供するファイルシステムとは独立した，TPBroker 専用のファイルシステムです。

キャラクタ型スペシャルファイル上に TPBroker ファイルシステムを作成します。TPBroker の回復に必要な情報を，この TPBroker ファイルシステム上に作成できます。

(2) OS が提供するファイルシステム

TPBroker ファイルシステムを使用すると高い信頼性を得ることができますが，一つのディスク領域を占有することになります。ディスクを効率良く使用したい場合は，OS が提供するファイルシステムを使用します。TPBroker は，デフォルトでは OS が提供するファイルシステムを使用します。

7.6.2 TPBroker ファイルシステムの作成方法

TPBroker ファイルシステムは，tsmkfs コマンドを使用して，キャラクタ型スペシャルファイル上に作成します。TPBroker ファイルシステムを使用する場合は，環境変数 TPFS または ADMFS に TPBroker ファイルシステムを作成したキャラクタ型スペシャルファイルを設定します。

7.6.3 TPBroker ファイルシステムの運用

TPBroker は，環境変数 TPFS または ADMFS にキャラクタ型スペシャルファイルが設定されている場合，TPBroker ファイルシステム上にファイルを作成します。このファイルは，TPBroker のデーモンおよびアプリケーションプログラムがアクセスします。このため，TPBroker ファイルシステムを作成したキャラクタ型スペシャルファイルには，TPBroker の運用者（TPBroker の開始および終了を行うユーザ），および TPBroker の使用者（ユーザアプリケーションを実行するユーザ）がアクセスできる必要があります。

7.7　トランザクショントレースの運用（UNIX）

ここでは，トレースファイルの詳細およびトランザクショントレース定義の変更など，トランザクショントレースの運用について説明します。

トランザクショントレースファイル以外の障害情報については，「10.3　障害の解決に必要な情報」を参照してください。

なお，トランザクショントレースの運用は UNIX 版でだけ有効です。

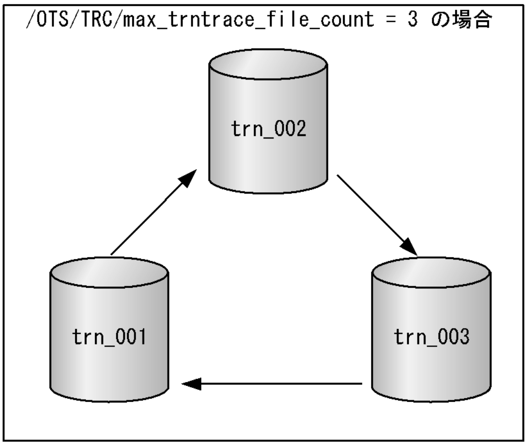
7.7.1　トレースファイル

トランザクショントレースは，OTS デーモンが作成した共用メモリにいったん格納され，その後トレースファイルへ記録されます。トレースファイルが存在しない場合，新規に作成されます。OTS デーモンが作成する共用メモリのサイズは約 512 キロバイトです。

トレースファイルは，trn_xxx（xxx：001 ～トランザクショントレース定義で設定した値）のファイル名で出力されます。トレースファイルがトランザクショントレース定義に設定したファイルサイズを超えると，新規にトレースファイルを作成し，以降のトレース情報は新しいファイルに記録されます。この場合，古いトレースファイルは世代管理され，バックアップとして残されます。世代数はトランザクショントレース定義で設定してください。

トレースファイルの世代管理の流れを次の図に示します。

図 7-1　トレースファイルの世代管理



説明

1. trn_001 にトランザクショントレースが記録される。
2. trn_001 のファイルサイズがトランザクショントレース定義に設定した値を超えると，新規にトレースファイル（trn_002）が作成される。

3. trn_001 はバックアップファイルになり，以降のトレース情報は trn_002 に記録される。
4. trn_002 のファイルサイズがトランザクショントレース定義に設定した値を超えると，新規にトレースファイル（trn_003）が作成される。
5. trn_002 はバックアップファイルになり，以降のトレース情報は trn_003 に記録される。
6. trn_003 のファイルサイズがトランザクショントレース定義に設定した値を超えると，trn_001 を初期化する。
7. trn_003 はバックアップファイルになり，以降のトレース情報は trn_001 に記録される。※
   注※
   トランザクショントレース定義に設定したトレースファイルの世代数を超えると，trn_001（最も古いバックアップファイル）を初期化して trn_001 の先頭からトレース情報を記録します。

トレースファイルは，OTS デーモンが起動（正常起動または再起動）するたびに，前回記録していたファイルの次の世代のファイルから記録を開始します。記録するファイルを決定するために，最新ファイルを検索します。検索は，ファイルの先頭部分（ファイルの世代データなど）だけをチェックするため，ファイルサイズは性能に影響しません。ただし，存在するすべてのトレースファイルがチェック対象のため，ファイル数は性能に影響します。トランザクショントレース定義 /OTS/TRC/max_trntrace_file_count の値を設定するときは，このことを考慮してください。

トレースファイルのファイルサイズがトランザクショントレース定義 /OTS/TRC/max_trntrace_file_size に設定したサイズを超過した場合，次のファイルに切り替わって記録を続けます。切り替えるファイルがすでに存在していたときは，ファイルを初期化し，ファイルの先頭から記録します。

記録しているトレースファイルが削除された場合，トレースデーモンはそのファイルに記録し続けているように動作します。その間に運用中のトランザクショントレースデータは失われます。ただし，トレース情報がトランザクショントレース定義 /OTS/TRC/max_trntrace_file_size に設定したサイズを超過してトレースファイルが次のファイルに切り替わるタイミングで，トレースファイルへの記録が再開始されます。

7.7.2 トレースファイルの出力先

トランザクショントレースファイルは，trn_xxx（xxx：001 ～トランザクショントレース定義で設定した値）のファイル名で次のどれかに出力されます。

- $TPSPOOL/trcinfo/trnspool/dcopltrc/
  デフォルトではこのディレクトリに出力されます。
- $TPB_TRN_TRACE_PATH/trnspool/dcopltrc/
  環境変数 TPB_TRN_TRACE_PATH を設定した場合に出力されます。このディレクト

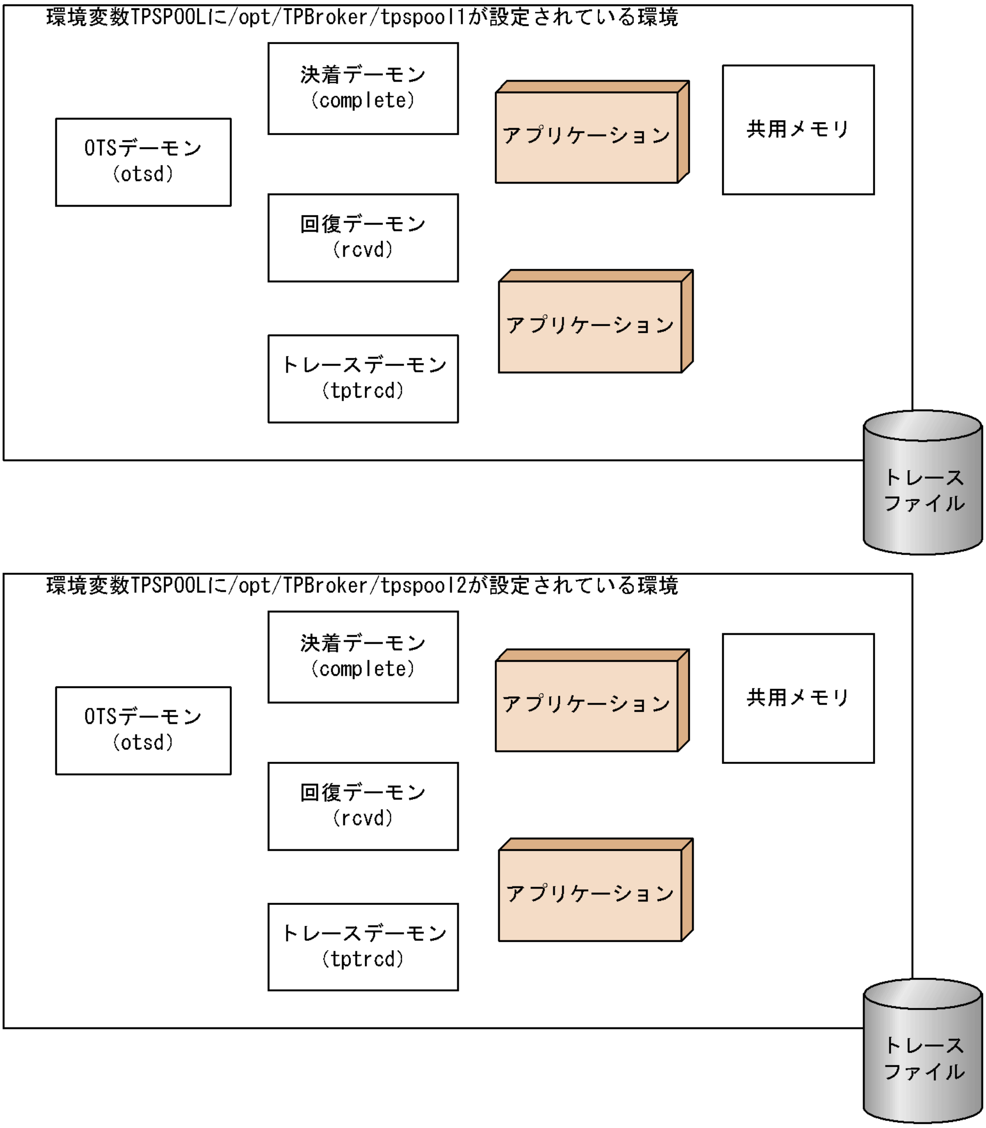
リは tssetup コマンド実行時に作成されます。tssetup コマンド実行後にトレースファイルの出力先を変更したい場合，TPBroker を終了した状態で環境変数 TPB_TRN_TRACE_PATH に適切なディレクトリを設定し，設定したディレクトリを手動で作成してください。環境変数 TPB_TRN_TRACE_PATH で設定したディレクトリとその中のファイルは，-d オプションを付けた tssetup コマンドで削除できます。$TPB_TRN_TRACE_PATH/trnspool/dcopltrc/ は TPBroker を再開始すると自動的に作成されますので注意してください。

- 標準出力
  tsedtrntrc コマンドを実行した場合，上記二つのどちらかに出力されたトレースファイルの情報を標準出力に出力します。

7.7.3 トレース取得範囲

トレースファイルは OTS デーモンが管理する環境のトレース情報出力ディレクトリに作成されます。一つの OTS デーモンに対して一つのトレースデーモンが起動されます。環境変数 TPSPOOL に設定されているディレクトリ下の環境が，トレースの取得範囲になります。トランザクショントレースがトレースを取得する範囲の例を次の図に示します。

図 7-2 トレース取得範囲



7.7.4 トランザクショントレース定義の変更

トランザクショントレースは，デフォルトではトランザクションブランチがロールバックした場合，またはヒューリスティック決着した場合にトレースを記録します。デフォルトの場合以外の情報を記録するには，トランザクショントレース定義を変更する必要があります。

トランザクショントレース定義を tsdefvalue コマンドで変更し，必要な情報をトレースに記録するように TPBroker に指示します。定義の変更は TPBroker を再開始すると有

効になります。なお，トランザクショントレースに記録する情報量を増やすと，システムのスループットが悪化するので注意してください。トランザクショントレース定義の詳細については，「8.3.7　トランザクショントレース定義」を参照してください。

7.7.5　注意事項

- TPBroker がダウンした場合，TPBroker の再開始時にトランザクションの回復処理を行います。その場合，トランザクション決着の結果が 2 回記録されることがあります。
- トップトランザクションブランチで rollback_only() を発行した場合でも，トップトランザクションブランチがリソースマネージャにアクセスしていなければ，リードオンリーのトランザクションブランチとして出力されることがあります。
- OTS Fast Path Option を使用している場合，サーバコール時にサーバプロセスが異常終了しても，トップトランザクションブランチがリソースマネージャにアクセスしていなければ，リードオンリーのトランザクションブランチとして出力されることがあります。
- 誤って TPBroker 運用中に tssetup コマンドに -d オプションを付けて実行した場合，共用メモリおよびメッセージキューが残ってしまいます。残ってしまった共用メモリおよびメッセージキューを ipcs コマンドで確認し，ipcrm コマンドで削除してください。
- トランザクショントレースは次の条件の場合，取得できないことがあります。
  - OTS デーモンが共用メモリの初期化に失敗した場合
  - トレースデーモンが起動していない場合（連続異常終了時）
  - トランザクションが動作するプロセス（該当プロセスだけ）で，共用メモリアクセスの初期化に失敗した場合
  - システムダウン後の再開始時に共用メモリへのアクセスが失敗した場合
  - システムダウン時にロールバックしたトランザクションブランチの場合

7.8　TPBroker デーモン

ここでは，TPBroker のデーモンプロセスと各プロセスの機能について説明します。

7.8.1　TPBroker のデーモンプロセス

ORB を除く TPBroker のプロセス構造と各プロセスの機能を次の表に示します。

表 7-2　TPBroker のプロセス構造と各プロセスの機能

名称	プロセス名	個数	機能
OTSD（OTS デーモン）	otsd	1（OTS 稼働ノード）	アプリケーションプログラムの監視，全面回復
COMPD（決着デーモン）	complete	1 ～ n（OTS 稼働ノード）	トランザクション決着処理（正常時，回復時）
RCVD（回復デーモン）	rcvd	1 ～ n（OTS 稼働ノード）	トランザクション部分回復（ルートブランチ）
TCS（トランザクションコンテキストサーバ）	trnctxsv	1（OTS 稼働ノード）	トランザクションコンテキスト管理
ADMD（ADM デーモン）	admd	• UNIX 版 1 ～ 5（各サーバノード） • Windows 版 1 ～ n（各サーバノード）	プロセス制御
TRCD（トレースデーモン）※	tptrcd	1（OTS 稼働ノード）	トランザクショントレース

注※
　TRCD（トレースデーモン）は，UNIX 版でだけ有効です。

各プロセスの内容を次に示します。

● OTSD（OTS デーモン）
OTSD は，トランザクションに参加しているすべてのユーザプロセスおよびシステムプロセス（COMPD，RCVD）を監視します。監視対象のプロセスが異常終了した場合，適切な処置を行います。
COMPD および RCVD が，それぞれ 3 分間に 45 回を超えて連続異常終了した場合は，OTSD が停止します。

● COMPD（決着デーモン）
COMPD は，トランザクションの決着処理を行います。COMPD では 2 相コミットプロトコルを実現しています。トランザクションオリジネータがコミットまたはロールバックでトランザクションの終了要求をした場合，OTS は COMPD とコンタクトを取ります。

● RCVD（回復デーモン）

RCVD は，異常終了したトランザクションの状態を回復します。異常終了する前のトランザクションの状態によって，トランザクションのコミットまたはロールバックをします。

● TCS（トランザクションコンテキストサーバ）

TCS は，TPBroker のために，トランザクションコンテキストを作成および管理します。TCS の詳細については，「5.　Java OTS 機能（Java）」を参照してください。

● ADMD（ADM デーモン）

ADMD は，OTS と ORB を統合して運用するための機能を提供しています。サンプルのプロセス監視定義ファイル（$TPDIR/adm/admconf.cf）では，ADMD は OTSD と VisiBroker のスマートエージェントである OSAgent のプロセスを監視します。ただし，Visual Studio 2005 対応 Windows 版の ADM は，VisiBroker for C++ のスマートエージェントである OSAgent のプロセスのみ監視します。

● TRCD（トレースデーモン）

TRCD は，トランザクショントレースのファイルへの出力およびトランザクショントレースファイルの管理をします。TRCD は，UNIX 版でだけ有効です。

なお，ORB のデーモンについては，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

7.9　TPBroker のバージョンアップ

TPBroker のバージョンアップ時には，アプリケーションプログラムの移行，起動プロパティの変更が必要となります。詳細については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

8 定義

TPBroker では，定義によって実行環境ごとにシステムをカスタマイズできます。この章では，TPBroker の定義の設定方法，詳細，および定義例について説明します。

8.1　定義の概要

TPBroker では，定義によって実行環境ごとにシステムをカスタマイズできます。

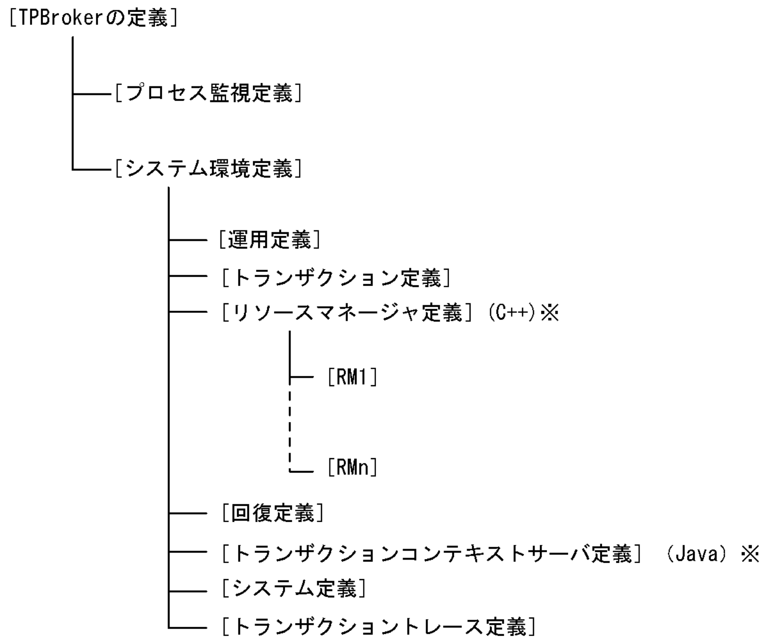
8.1.1　定義体系

TPBroker の定義は，プロセス監視定義とシステム環境定義に分けられます。

また，システム環境定義の定義体系は，運用系，トランザクション系，リソースマネージャ系（C++），回復系，トランザクションコンテキストサーバ系（Java），システム系，およびトランザクショントレース系に分かれています。

TPBroker の定義体系を次の図に示します。

図 8-1　定義体系



RM1，...，RMn は，リソースマネージャの名称です。リソースマネージャごとの設定が必要な場合に使用します。

注※

リソースマネージャ定義およびトランザクションコンテキストサーバ定義は，Cosminexus TPBroker ではサポートしていません。

8.1.2　定義情報の設定

(1)　プロセス監視定義の設定方法

プロセス監視定義はプロセス監視定義ファイルを使用して設定します。プロセス監視定義ファイルは，メモ帳などのテキストエディタを使用して編集してください。

TPBroker の運用支援機能の動作中にプロセス監視定義ファイルを変更した場合，変更を反映するには，次のどちらかを実施する必要があります。

- admreload コマンドでプロセス監視定義ファイルの再読み込みを実行する。
- admstop コマンドで TPBroker を正常終了させてから，admstart コマンドで再度 TPBroker を開始する。

(2)　システム環境定義の設定方法

システム環境定義は定義設定コマンドを使用して設定します。

tssetup コマンドを使用して実行環境を作成すると，定義情報にデフォルト値が設定されます。このデフォルト値に対して，TPBroker の定義設定コマンドを使用して，リソースマネージャに関する定義情報を設定したり，デフォルト値を変更したりできます。

システム環境定義を設定および変更するときは，次の点に注意してください。

- tssetup コマンド実行時にデフォルトで設定されるシステム環境定義の定義キー（/OTS，/ADM，/SYSTEM など）および定義パラメタは，削除しないでください。
- tssetup コマンドを実行すると，それまで使用していたシステム環境定義の変更および追加内容は，上書きまたは削除され，定義内容はすべてデフォルト値になります。
- システム環境定義の運用定義（定義キーが /ADM の定義）の変更は，次に運用支援機能実行環境を開始したときに有効になります。tsdefvalue コマンドでシステム環境定義を変更した場合，admstop コマンドで運用支援機能実行環境を正常終了させてから，admstart コマンドで再度運用支援機能実行環境を開始してください。

定義設定コマンドの詳細は，「9.3　運用コマンドの詳細」を参照してください。

8.2 プロセス監視定義の詳細

ここでは，プロセス監視定義のフォーマットや定義項目の詳細などについて説明します。

8.2.1 定義項目

プロセス監視定義ファイルには次の項目を設定できます。

- 起動（監視対象）プロセスの識別子
- 起動（監視対象）プロセスのファイル名やコマンドなど
  正常起動，再起動，および強制正常起動についてそれぞれ設定できます。
- プロセス起動用コマンドおよびプロセス停止用コマンドのタイムアウト値
- プロセス起動に失敗した場合の TPBroker の処置
  次のどれか一つを設定します。
  - TPBroker の運用支援機能を終了する
  - 開始に失敗したプロセスを再起動する
  - 何もしない
- 監視対象プロセスが異常終了した場合の TPBroker の処置
  次のどれか一つを設定します。
  - TPBroker の運用支援機能を終了する
  - 異常終了したプロセスを再起動する
  - 何もしない
  - 特定のコマンドを発行する
- プロセスを起動するタイミング
- 一定時間内に連続して異常終了する回数の上限
- 正常停止用コマンド
- 強制停止用コマンド
- コマンド実行用のユーザ ID
- コマンド実行用のグループ ID
- プロセス起動時に設定する環境変数名と値
- TPBroker 開始時に設定する環境変数名と値

以降で，定義項目の詳細を説明します。

8.2.2 プロセス監視定義のフォーマット

プロセス監視定義の記述フォーマットを次の図に示します。

図中の 1 ～ 18 の数字は，「8.2.4　定義項目の詳細」の表 8-1 の項番と対応しています。

図 8-2 プロセス監視定義の記述フォーマット

```
#これはコメントです。
18
SampleServer :"C:¥sample¥SampleServer.exe"  : :"C:¥sample¥SampleServer.exe  -f":300,600 :¥
1            2                              3 4                               5    17
none :restart : :none:5:"C:¥sample¥StopServer.exe"  :¥
6     7      8 9  10 11
"C:¥sample¥StopServer.exe"  :1234:5678:"OSAGENT PORT=14022","AAAA=aaaa"
12                            13  14  15
PUTENV:BBBB=bbbb
16
PUTENV:CCCC=cccc
.......
```

図中に記載のフォーマットは Windows 表記となっています。UNIX 上では UNIX 表記に置き換えてください。

プロセス監視定義ファイルのサンプルとして $TPDIR/adm/admconf.cf を提供しています。

8.2.3 プロセス監視定義の記述規則

(1) 共通規則

- 設定項目は，「:」（コロン）で区切ります。
- 省略する場合はスペースを入れてください。
- 1 行には 1023 文字まで記述できます。

(2) プロセス名およびコマンド名に関する規則

- 「"」（引用符）で囲んで指定してください。
- 「"」で囲んでいる中に「"」が含まれる場合は，次の例の下線部分のように，囲んだ内側の「"」の前に「¥」を指定してください。

  例 `"/opt/TPBrokerV5/bin/xxxx¥"abcd¥" :-f"`

- ドライブ名を含む絶対パス名で指定してください。

  例 （誤）`"¥tpbroker¥examples¥library¥libsrv.exe"`
  　（正）`"C:¥tpbroker¥examples¥library¥libsrv.exe"`

  なお，「|」（パイプ），「>」および「>>」（リダイレクト）を含んだパスを指定すると正しく動作しません。
- 「"」「¥」「$」を含むファイル名やディレクトリ名を指定しないでください。
- スペースを含むパス名は「"」で囲んでください。定義項目全体を「"」で囲んでいる場合は，次の例の下線部分のように，パス名を囲む「"」の前に「¥」を指定してください。

  例 `～:"¥"C:¥Program Files¥test¥process.exe¥" option1 option2":～`

- Windows 版でバッチファイルを指定する場合，cmd.exe を使用してください。
  例　　"C:¥WINNT¥SYSTEM32¥cmd.exe /c C:¥Sample¥testprc.bat"

- Windows 版の場合，ネットワークドライブ上のコマンドやプロセスを指定しないでください。

(3) プロセス起動用コマンドおよびプロセス停止用コマンドのタイムアウト値に関する規則

- プロセス起動コマンド及び停止コマンドのタイムアウト値に関しては以下に従ってください。なお，停止用コマンドのタイムアウト値は，TPBroker 05-15 以降で有効です。
- プロセス起動用コマンドのタイムアウトとプロセス停止用タイムアウト値を「,」(コンマ) で区切って指定してください。
  例　　次の場合，プロセス起動用コマンドのタイムアウト値に 150 秒，プロセス停止用タイムアウト値に 600 秒が設定されます。
  ～ :150,600: ～
- プロセス起動用コマンドのタイムアウト値を省略した場合，デフォルト値 300 秒が設定されます。
  例　　次の場合，プロセス起動用コマンドのタイムアウト値にはデフォルト値 300 秒，プロセス停止用コマンドのタイムアウト値に 600 秒が設定されます。
  ～ :,600: ～
- プロセス停止用コマンドのタイムアウト値を省略した場合，デフォルト値 300 秒が設定されます。
  例　　次の場合，プロセス起動用コマンドのタイムアウト値に 150 秒，プロセス停止用コマンドのタイムアウト値にデフォルト値 300 秒が設定されます。
  ～ :150,: ～
  「,」(コンマ) は省略可能です。
- 「,」(コンマ) のみを指定した場合，プロセス起動用コマンドおよびプロセス停止用コマンドのタイムアウト値には，デフォルト値 300 秒が設定されます。
  例　　～ :,: ～

(4) プロセス起動時に設定する環境変数名と値に関する規則

- 「"」で囲み，複数ある場合は，「,」（コンマ）で区切って指定してください。
- 「"」および「,」の前後には，スペースを含めないでください。
- 改行はしないでください。
- 「"」で囲んだ中にコメントを記述しないでください。
  例　　（誤）:"OSAGENT_PORT=#comment
  　　　　　14000"

(5) TPBroker 起動時に設定する環境変数名と値に関する規則

- この定義項目（PUTENV: 環境変数名 = 値）は，プロセス監視定義の最後に指定してください。この定義項目のあとに，さらに定義項目を指定すると，定義エラーになり

ます。

- 2 個目以降の PUTENV: のあとに空白またはタブだけを記述した場合，定義エラーになります。

例　　（誤）PUTENV:OSAGENT_PORT=14001
```
        PUTENV:
```

- 同じ環境変数を指定した場合，あとから指定した環境変数が有効になります。

例　次の場合，OSAGENT_PORT には 14003 が設定されます。

```
PUTENV:OSAGENT_PORT=14002
PUTENV:OSAGENT_PORT=14003
```

環境変数 PATH に複数のパスを設定するような場合は，次のように「:」で区切って指定してください。

例　`PUTENV:PATH=/opt/TPBroker/bin:/usr/java130/bin`

- 環境変数の値にスペースが含まれる場合でも，値を「"」で囲まないでください。

(6)　改行文字に関する規則

- 一つの設定項目を複数行に分けて記述することはできません。

例　（誤）"/opt/TPBrokerV5/bin/otsd ￥
```
                   -f"
```

8.2.4　定義項目の詳細

プロセス監視定義の項目の内容を次の表に示します。プロセス監視定義の指定項目内容の説明では，例の記述が Windows 表記となっています。UNIX 上では UNIX 表記に置き換えてください。

表 8-1　プロセス監視定義の項目の内容

項番	定義項目	指定する値	項目の説明
1	起動プロセスの識別子 **（必須）**	**直接起動方式の場合** 先頭が a ～ z または A ～ Z の英字で始まる，4 文字以上 32 文字以下の英数字 **間接起動方式の場合** 先頭が 0 ～ 9 の数字で始まる，4 文字以上 32 文字以下の英数字	－
2	起動する監視対象プロセスの名称－正常起動 **（必須）**	255 文字以内の文字列 **直接起動方式の場合** 起動するプロセス名 **間接起動方式の場合** 発行するコマンド名	－

項番	定義項目	指定する値	項目の説明
3	起動する監視対象プロセスの名称－再起動	255 文字以内の文字列 **間接起動方式の場合** プロセス ID 取得用のコマンド名	名称を指定しない場合は，項番 2 で指定した名称が適用されます。 間接起動方式で，コマンドが未設定の場合，TPBroker はプロセスを起動および停止するだけで，監視はしません。
4	起動する監視対象プロセスの名称－強制正常起動	255 文字以内の文字列	名称を指定しない場合は，項番 2 で指定した名称が適用されます。 間接起動方式の場合は，無視されます。
5	プロセス起動用コマンドおよびプロセス停止用コマンドのタイムアウト値※	0 ～ 1800（秒） デフォルト：300 「プロセス起動用コマンドのタイムアウト値 , プロセス停止用コマンドのタイムアウト値」の形式で指定します。	プロセス開始コマンドのタイムアウト値は間接起動方式の場合だけ有効です。 プロセス起動用コマンドのタイムアウトおよびプロセス停止用コマンドのタイムアウトに 0 を設定した場合，コマンドのプロセスを即時に強制終了します。 **プロセス起動用コマンドのタイムアウト** 項番 2 で指定したコマンドの終了を，プロセス起動用コマンドのタイムアウト値で指定した時間だけ待ちます。指定時間が経過してもコマンドが終了しない場合は，コマンドのプロセスを強制停止します。 **プロセス停止用コマンドのタイムアウト** 項番 11 および項番 12 で指定したコマンドの終了を，プロセス停止用コマンドのタイムアウト値で指定した時間だけ待ちます。指定時間が経過してもコマンドが終了しない場合は，コマンドのプロセスを強制停止します。
6	プロセス起動に失敗した場合の TPBroker の処置 **（必須）**	次のどれか一つを指定します。 • down TPBroker の運用支援機能を終了する • restart 起動に失敗したプロセスを再起動する • none 何もしないで処理を続ける デフォルト：none	「restart」を指定した場合は，プロセス起動を連続 3 回失敗すると，それ以降は再起動処理を行いません。

項番	定義項目	指定する値	項目の説明
7	監視対象プロセスが異常終了した場合のTPBrokerの処置 **（必須）**	次のどれか一つを指定します。 • down TPBrokerの運用支援機能を終了する • restart 異常終了したプロセスを再起動する • none 何もしない • command 特定のコマンドを発行する デフォルト：none	「command」を指定した場合は，発行するコマンド名を指定します（項番8を参照のこと）。
8	監視対象プロセスが異常終了した場合に発行するコマンド名	255文字以内の文字列	項番7で「command」を指定した場合，必ず指定します。
9	プロセスを起動するタイミング **（必須）**	次のどちらかを指定します。 • none admstartコマンド発行時に起動 • command admstartprcコマンド発行時に起動 デフォルト：none	－
10	一定時間内に連続して異常終了する回数の上限 **（必須）**	0以上の整数 デフォルト：3	監視時間は10分です。 指定値回目の再起動を抑止します。指定値0と1は再起動を行いません。 この定義の指定は，項番7で「restart」を指定している場合だけ有効です。
11	正常停止用コマンド	255文字以内の文字列	コマンドを指定しない場合は，システムコール（UNIXの場合はkill()，Windowsの場合はTerminateProcess()）でプロセスが直接停止します。
12	強制停止用コマンド	255文字以内の文字列	コマンドを指定しない場合は，項番11に指定したコマンドが発行されます。
13	コマンド実行用のユーザID	0～59999の符号なし整数	Windows版の場合は無視されます。
14	コマンド実行用のグループID	0～59999の符号なし整数	Windows版の場合は無視されます。
15	プロセス起動時に設定する環境変数名と値	511文字以内の文字列 「"環境変数名＝値"，"環境変数名＝値"…」の形式で指定します。全体で511文字以内にしてください。	必要な場合に指定します。

項番	定義項目	指定する値	項目の説明
16	TPBroker 開始時に設定する環境変数名と値	511 文字以内の文字列 「PUTENV：環境変数名 = 値」の形式で指定します。 ADMFS，ADMSPOOL，TPDIR，TPFS，TPDIR は設定せず，admsetup コマンド実行時の環境変数を引き継がせてください。	必要な場合に指定します。 環境変数は，0 ～ 100 個まで，1 行に 1 個だけ指定できます。 ここで指定した環境変数は，監視対象プロセスすべてに設定されます。 項番 15 で指定した環境変数名と重複する場合，プロセス起動時には項番 15 の値が設定されます。 指定しない場合，admsetup コマンド実行時の環境変数が使用されます。
17	改行文字	¥	定義が複数行にわたる場合は，必ず指定します。
18	コメント	#	「#」から行末までをコメントとみなします。 例 1 　# コメント 1 　aaaa:"C:¥tpbroker¥bin¥xxxx": .... 例 2 　aaab:"C:¥tpbroker¥bin¥xxxx": ¥ # コメント 2

（凡例）

－：該当する内容がないことを表します。

注※

停止用コマンドのタイムアウト値は，TPBroker 05-15 以降で有効です。

8.3 システム環境定義の詳細

システム環境定義の定義項目を次の表に示します。この定義項目は，実行環境をルートとするパスで表現されます。

表 8-2 システム環境定義の定義項目

定義の種類	定義名称	定義パス名※1	デフォルト値	値の範囲	指定値タイプ	引き継ぎ※2
運用定義	開始モード	/ADM/set_conf_mode※5	"MANUAL"（UNIX）"AUTO"（Windows，ただし，定義値が値の範囲にない場合は ”MANUAL ”）	"MANUAL"|"MANUAL2"|"MANUAL3"|"MANUAL4""AUTO"	文字列	×
	ポート番号	/ADM/port_id_info※5	20058	1024 〜 65535	整数	×
	ログ世代管理数	/ADM/backup_count※5	3	3 〜 10	整数	×
	最大監視対象プロセス数	/ADM/max_process_num※5	100	1 〜 4096	整数	○
	ADM 停止モード（Windows）	/ADM/service_stop_mode	"FORCE"	"FORCE"|"FORCE_RECOVER"	文字列	×
	並列実行モード	/ADM/set_parallel_mode	"N"	"Y"|"N"	文字列	○
	並列実行数	/ADM/set_parallel_count	5	1 〜 64	整数	○
	リダイレクトモード（Windows）	/ADM/set_redirect_mode	"N"	"Y"|"N"	文字列	×
	リダイレクトファイル名（Windows）	/ADM/set_redirect_filename	""	文字列	文字列	×
	リダイレクトファイルサイズ（Windows）	/ADM/set_redirect_filesize	1024	1024 〜 65535（キロバイト）	整数	×

定義の種類	定義名称	定義パス名※1	デフォルト値	値の範囲	指定値タイプ	引き継ぎ※2
トランザクション定義	同時実行ブランチ数	/OTS/TM/process_count※5	32 256 （Cosminexus TPBroker）	1 ～ 4096	整数	○
	最大接続可能CRM ブランチ数	/OTS/TM/max_crm_branch_count※5	8	0 ～ 256	整数	○
	回復プロセス数	/OTS/recovery_process_count※5	1	1 ～ 128	整数	○
	決着プロセス数	/OTS/completion_process_count※5	1	1 ～ 128	整数	○
	決着プロセスポート番号ベース	/OTS/completion_process_port_base※5	20085	1024 ～ 32767	整数	○
	トレースファイル削除インタバル	/OTS/set_trace_remove_interval※5	43200	0 ～ 2147483647（秒）	整数	○
	トレースファイル最大残存数	/OTS/max_trace_remain_num※5	500	0 ～ 2147483647（ファイル）	整数	○
	回復プロセス環境変数	/OTS/recovery_process_env	なし	文字列の配列	文字列配列	×
	決着プロセス環境変数	/OTS/completion_process_env	なし	文字列の配列	文字列配列	×
	決着プロセスホスト名	/OTS/completion_process_ipaddr_info	""	文字列	文字列	○
	デーモンプロセスホスト名	/OTS/set_ipaddr_info	""	文字列	文字列	×
	OTS 監視プロセス最大数（C++）	/OTS/max_process_monitor_count	300	300 ～ 30000	整数	○
	トランザクションステータス書き込みモード	/OTS/TM/set_status_write_mode	"none"	"none" | "immediate"	文字列	×

定義の種類	定義名称	定義パス名※ 1	デフォルト値	値の範囲	指定値タイプ	引き継ぎ※ 2
	トランザクション引き継ぎモード	/OTS/TM/set_recovery_mode	0 1 （Cosminexus TPBroker）	0|1	整数	○
	コンパイラ選択（C++）	/OTS/set_compiler_info	"MS80"（WindowsVisual Studio 2005 の場合），"SWCC"（Solaris の場合），"GLNX"（Linux AS4 の場合） "AIXC"（AIX の場合）	"MS80"（WindowsVisual Studio 2005 の場合），"SWCC"（Solaris の場合），"GLNX"|"GL32"（Linux AS4 の場合） "AIXC"|"XLC8"|"XLC9"（AIX の場合）	文字列	×
	tsstart コマンドタイムアウト値	/OTS/tsstart_timeout	300	300 ～ 1800（秒）	整数	×
リソースマネージャ定義（C++）※ 3	非同期インタバル	/OTS/RM/set_xa_async_interval※ 5	1000	1 ～ 1000000（ミリ秒）	整数	×
	オープン文字列	/OTS/RM/RMn/xa_open_string_info	""	文字列	文字列	○※ 4
	クローズ文字列	/OTS/RM/RMn/xa_close_string_info	""	文字列	文字列	○※ 4
	オープン文字列	/OTS/RM/RMn/DMN/xa_open_string_info	""	文字列	文字列	○※ 4
	クローズ文字列	/OTS/RM/RMn/DMN/xa_close_string_info	""	文字列	文字列	○※ 4
	オープン文字列	/OTS/RM/RMn/UAPn/xa_open_string_info	""	文字列	文字列	×

定義の種類	定義名称	定義パス名※1	デフォルト値	値の範囲	指定値タイプ	引き継ぎ※2
	クローズ文字列	/OTS/RM/RMn/UAPn/xa_close_string_info	""	文字列	文字列	×
	xa_open 発行タイミング	/OTS/RM/set_xa_open_timing	"deferred"	"deferred"|"resolve"	文字列	×
	xa_open 発行単位	/OTS/RM/set_xa_open_scope /OTS/RM/RMn/set_xa_open_scope	"process"	"process"|"thread"	文字列	○
	RMERR 動作モード	/OTS/RM/RMn/set_xa_rmerr_action	"abort" "retry1"（Cosminexus TPBroker）	"abort"|"retry1"	文字列	○
	RMFAIL 動作モード	/OTS/RM/RMn/set_xa_rmfail_action	"abort" "retry1"（Cosminexus TPBroker）	"abort"|"retry1"	文字列	○
回復定義	回復インタバル	/OTS/RCV/set_retry_time※5	10	1 ～ 65535（秒）	整数	○
	トランザクション回復タイミング	/OTS/RCV/set_startup_recovery_skip	0 1（Cosminexus TPBroker）	0|1	整数	○
トランザクションコンテキストサーバ定義（Java）※3	TCS 名	/OTS/TCS/trn_ctx_sv_name	""	0 ～ 16 文字	文字列	×
	トランザクションデフォルトタイムアウト値	/OTS/TCS/transaction_default_timeout	180	0 ～ 2147483647（秒）	整数	×

定義の種類	定義名称	定義パス名※1	デフォルト値	値の範囲	指定値タイプ	引き継ぎ※2
システム定義	システム識別子情報	/SYSTEM/system_id_info	""	4 文字	文字列	×
トランザクションシントレース定義	トランザクションブランチの決着状況	/OTS/TRC/set_trntrace_level	"err"	"err"|"all"|"none"	文字列	○
	トレースファイルサイズ	/OTS/TRC/max_trntrace_file_size	1024	1024 〜 1048576（キロバイト）	整数	○
	トレースファイル世代数	/OTS/TRC/max_trntrace_file_count	3	3 〜 256	整数	○

注※ 1

定義パス名には定義キーと定義パラメタが含まれています。

- **定義キー**：定義項目を区分けして格納しておく位置を識別する名称
  表 8-2 では ADM や RM などに相当します。
  定義キーは，英字から始まり，英数字，「_」（アンダースコア）で構成される半角文字列です。これ以外の文字で構成される名称を使用した場合，定義が無効になる場合があります。
- **定義パラメタ**：定義項目の名称
  表 8-2 では set_conf_mode や process_count などに相当します。
  定義パラメタは，英字から始まり，英数字，「_」（アンダースコア）で構成される 1 〜 32 文字の半角文字列です。これ以外の文字で構成される名称を使用した場合，定義が無効になる場合があります。

注※ 2

再開始したときに，前回開始時に使用したシステム環境定義を引き継ぐかどうかを示します。
○：引き継ぐ
×：引き継がない

注※ 3

Cosminexus TPBroker では，サポートしていません。

注※ 4

回復デーモンだけ再開始したときに引き継ぎます。決着デーモンは引き継ぎません。

注※ 5

システム環境定義を削除しないでください。

8.3.1 運用定義

開始モード /ADM/set_conf_mode "MANUAL"｜"MANUAL2"｜"MANUAL3"｜"MANUAL4"｜"AUTO"

〜 <<"MANUAL"（UNIX），"AUTO"（Windows）>>

TPBroker のシステムの開始方法を指定します。

- "MANUAL"
  手動開始になります。異常終了後は自動開始になります。
- "MANUAL2"
  手動開始になります。異常終了後も手動開始になります。
- “ MANUAL3 ”※ 1
  手動開始になります。異常終了後は自動開始になります。
  TPBroker が開始している状態の時に，OS が異常終了を起こしても次回，TPBroker が起動したときは手動開始になります。
  この値は，TPBroker 05-15 以降で有効です。
- "MANUAL4"
  手動開始になります。異常終了後は自動開始になります。
  TPBroker が開始している状態の時に，OS が異常終了やシャットダウンを起こしても次回，TPBroker が起動したときは手動開始になります。
  この値は，TPBroker 05-15 以降で有効な指定です。
- "AUTO"
  自動開始になります。

注※ 1

Windows 版では開始モード MANUAL3 をサポートしていません。

Windows 版の場合，TPBroker は OS シャットダウン時には正常に終了され，次回の TPBroker 開始時には正常開始されるためです。

デフォルト値は，UNIX 版の場合は "MANUAL"，Windows 版の場合は "AUTO" です。

範囲外の値を指定した場合は，"MANUAL" を適用します。

例 `tsdefvalue /ADM set_conf_mode -s "MANUAL"`

ポート番号 /ADM/port_id_info

〜 ((1024 〜 65535))<<20058>>

TPBroker の運用支援機能が提供するデーモンプロセスと運用コマンド間の通信で使用するポート番号を指定します。指定されたポート番号がすでにほかのシステムで使用されている場合は，OS によって割り当てられたポート番号で動作し，割り当てられたポート番号をこのシステム環境定義に自動的に設定します。

例 `tsdefvalue /ADM port_id_info -i 20058`

この定義の値は，TPBroker が OS に登録されていない状態で変更してください。また，OS に複数の TPBroker を登録する場合は，それぞれの環境でこの定義の値が重ならないように指定してください。
TPBroker の運用支援機能の動作中に，この定義の値を変更しないでください。また，UNIX 版の場合，admsetup コマンド実行後にこの値を変更しないでください。変更した場合は，admstart，admstop コマンドで TPBroker を開始および終了できません。

ログ世代管理数　/ADM/backup_count

〜 ((3 〜 10))<<3>>

TPBroker の運用支援機能が使用する環境変数 ADMSPOOL で設定される作業ディレクトリの世代管理数を指定します。プロセス監視機能の異常終了後の再開始時に自動的に退避する作業ディレクトリの最大数です。

例　`tsdefvalue /ADM backup_count -i 3`

最大監視対象プロセス数　/ADM/max_process_num

〜 ((1 〜 4096))<<100>>

TPBroker が監視できるプロセス数の最大値を指定します。

例　`tsdefvalue /ADM max_process_num -i 100`

ADM **停止モード**（Windows）　/ADM/service_stop_mode "FORCE"|"FORCE_RECOVER"

〜 <<"FORCE">>

この定義は Windows 版だけに提供される定義です。
サービス「TPBroker」を終了する場合に，-f または -fr オプションを付けた admstop コマンドで終了するかどうかを決定します。

- "FORCE"
  サービス「TPBroker」を終了する場合に，admstop -f で終了します。
- "FORCE_RECOVER"
  サービス「TPBroker」を終了する場合に，admstop -fr で終了します。

tssetup コマンド入力時には，デフォルトのシステム環境定義として "FORCE" が設定されます。また，このシステム環境定義は，再開始したときに引き継がれません。

例　`tsdefvalue /ADM service_stop_mode -s "FORCE"`

この定義が未設定の場合，および整数型が指定されている場合は "FORCE" が指定されているものとして動作します。
また，定義値が範囲外の場合，メッセージ KFCB29189-W をイベントログに出力します。メッセージの詳細は，「11.2　メッセージ一覧」を参照してください。

並列実行モード　/ADM/set_parallel_mode "Y"|"N"

〜 <<"N">>

admstart コマンドを入力する場合，admstop コマンドを入力する場合，および

TPBroker（ADM）のダウン後に再開始する場合に，ADM の監視対象プロセスを起動／停止するとき，ユーザが指定したグループごとに並列に起動／停止する機能を使用するかどうかを決定します。

- "Y"
  この機能を使用します。
- "N"
  この機能を使用しません。

この定義が未設定および "N" でも "Y" でもない場合は，"N" が設定されているものとして動作します。
この定義は，tssetup コマンド入力時のデフォルトのシステム環境定義としては設定されません。また，この定義は，再開始したときに引き継がれます。
例 `tsdefvalue /ADM set_parallel_mode -s "Y"`

並列実行数 /ADM/set_parallel_count

〜 ((1 〜 64))<<5>>
admstart コマンドを入力する場合，admstop コマンドを入力する場合，および TPBroker（ADM）のダウン後に再開始する場合の，並列実行時の同時起動または停止プロセス数を指定します。
この定義は /ADM/set_parallel_mode が "Y" の場合だけ有効になります。
また，admstartprc または admstopprc コマンドで -p オプションを指定して並列実行を行う場合には使用しません。
この定義が未設定および指定範囲外の場合，「5」が設定されているものとして動作します。
この定義は，tssetup コマンド入力時のデフォルトのシステム環境定義としては設定されません。また，この定義は，再開始したときに引き継がれます。
例 `tsdefvalue /ADM set_parallel_count -i 5`

リダイレクトモード（Windows） /ADM/set_redirect_mode "Y"|"N"

〜 <<"N">>
この定義は Windows 版だけに提供される定義です。
ADM デーモンの監視対象プロセスの標準出力，標準エラー出力をファイルに切り替えるかどうかを決定します。

- "Y"
  ファイルに切り替えます。コンソールに出力しません。
- "N"
  ファイルに切り替えません。コンソールに出力します。

例 `tsdefvalue /ADM set_redirect_mode -s "Y"`

リダイレクトファイル名（Windows） /ADM/set_redirect_filename

〜 < **文字列** ><<"">>
この定義は Windows 版だけに提供される定義です。
ADM デーモンの監視対象プロセスの標準出力，標準エラー出力を出力するファイル

名を絶対パスで指定します。世代管理をするため，ファイル名のあとに「1」または「2」が付加されます。この定義を省略すると，%ADMSPOOL%¥log¥stdlog1 と %ADMSPOOL%¥log¥stdlog2 というファイルが作成されます。このファイル名を格納するディレクトリはあらかじめ作成しておく必要があります。

例　`tsdefvalue /ADM set_redirect_filename -s "C:¥user1¥log.txt"`

リダイレクトファイルサイズ（Windows）　/ADM/set_redirect_filesize

〜 ((1024 〜 65535))<<1024>>(**単位：キロバイト**)

この定義は Windows 版だけに提供される定義です。

ADM デーモンの監視対象プロセスの標準出力，標準エラー出力を出力するファイルのサイズをキロバイト単位で指定します。メッセージの出力内容によって，出力ファイル長がこの指定値を超える場合があります。

例　`tsdefvalue /ADM set_redirect_filesize -i 2048`

8.3.2　トランザクション定義

同時実行ブランチ数　/OTS/TM/process_count

〜 ((1 〜 4096))<<32，256（Cosminexus TPBroker）>>

同時に起動するトランザクションブランチの数を指定します。

トランザクションサービスを利用するプロセス数，回復プロセス数，および回復処理を待っているトランザクションブランチ数の総数を指定します。

トランザクションブランチを発生させたアプリケーションプログラムが異常終了したとき，回復プロセスがほかのトランザクションブランチの決着処理をしている間，このトランザクションブランチは回復処理を待っている状態になります。この状態が長く続くと，起動できるトランザクションブランチ数が少なくなる場合があります。このため，異常終了後にトランザクションブランチを発生できるアプリケーションプログラムの扱いや，異常終了の頻度などを考慮してトランザクションブランチ数を指定する必要があります。指定数の目安を次に示します。

```
（トランザクションを実行するアプリケーションプログラム＋回復プロセス数）
   ＜（指定するトランザクションブランチ数）
   ＜＝（トランザクションを実行するアプリケーションプログラム×2
   ＋回復プロセス数）
```

なお，指定数が多くなればなるほどメモリ資源の効率が悪くなるので注意してください。

例　`tsdefvalue /OTS/TM process_count -i 32`

最大接続可能 CRM ブランチ数　/OTS/TM/max_crm_branch_count

〜 ((0 〜 256))<<8>>

一つのトランザクションブランチから生成するトランザクションブランチの最大数を指定します。ネスティッドトランザクションを実行する場合は，そのトランザクションブランチで開始するサブトランザクションも含めた数を指定します。

CosTransactions::Coordinator::register_resource および

CosTransactions::Coordinator::register_subtran_aware 関数を発行する場合は，そ

のトランザクションブランチでのこれらの関数の発行回数も含めて指定します。

例 `tsdefvalue /OTS/TM max_crm_branch_count -i 8`

回復プロセス数 /OTS/recovery_process_count

〜 ((1 〜 128))<<1>>

トランザクションブランチが異常終了したときに，トランザクションブランチの回復処理を並行して実行できる数を指定します。

トランザクションブランチが異常終了した場合は，ここで指定した数だけ並行してトランザクションブランチの回復処理をします。

例 `tsdefvalue /OTS recovery_process_count -i 1`

決着プロセス数 /OTS/completion_process_count

〜 ((1 〜 128))<<1>>

トランザクションブランチを決着するときに，決着処理を並行して実行できる数を指定します。

トランザクションブランチを決着する場合は，ここで指定した数だけ並行してトランザクションブランチの決着処理をします。

例 `tsdefvalue /OTS completion_process_count -i 1`

決着プロセスポート番号ベース /OTS/completion_process_port_base

〜 ((1024 〜 32767))<<20085>>

決着デーモンが使用するポート番号を指定します。ここで指定したポート番号から連続して，/OTS/completion_process_count で指定した値だけのポート番号を決着デーモンが使用します。

TPBroker を開始するときに，ここで指定したポート番号のポートをほかのプロセスが使用していると，決着デーモンが起動せず tsstart コマンドはタイムアウトします。この場合，この定義の値を変更してください。

例 `tsdefvalue /OTS completion_process_port_base -i 20085`

トレースファイル削除インタバル /OTS/set_trace_remove_interval

〜 ((0 〜 2147483647))<<43200>>(**単位：秒**)

$TPSPOOL/trace および $TPSPOOL/aptrace 下に作成されるトレースファイルを自動削除するときのインタバルを指定します。0 を指定した場合は，トレースファイルは自動削除されません。

例 `tsdefvalue /OTS set_trace_remove_interval -i 43200`

トレースファイル最大残存数 /OTS/max_trace_remain_num

〜 ((0 〜 2147483647))<<500>>(**単位：ファイル**)

$TPSPOOL/trace および $TPSPOOL/aptrace 下に作成されるトレースファイルを自動削除するとき，新しいファイルから，ここで指定したファイル数だけトレースファイルが残ります。0 を指定した場合は，すべてのトレースファイルが削除されます。ただし，使用中のトレースファイルは，自動削除の対象とはなりません。

例 `tsdefvalue /OTS max_trace_remain_num -i 500`

回復プロセス環境変数 /OTS/recovery_process_env

〜 < 文字列の配列 ><< なし >>

回復プロセスに追加設定する環境変数を「環境変数名 = 値」の形式で指定します。この定義は，必ず tsdefvalue コマンドの -a オプションで指定してください。複数の環境変数を指定する場合は，次のようにします。

```
tsdefvalue /OTS recovery_process_env -a "ENV1=VALUE1"¥
"ENV2=VALUE2" "ENV3=VALUE3"
```

指定できる定義の長さには制限があります。N 個の環境変数を指定する場合，次の条件を満たす必要があります。

（各文字列の長さの和）＋N＜＝ 2000

この定義にスペースが含まれる場合，スペース以降の環境変数は設定されません。次のように指定すると，「ENV3=VALUE3」は設定されません。

```
tsdefvalue /OTS recovery_process_env -a "ENV1=VALUE1" ""
"ENV3=VALUE3"
```

決着プロセス環境変数　/OTS/completion_process_env

〜 < 文字列の配列 ><< なし >>

決着プロセスに追加設定する環境変数を「環境変数名 = 値」の形式で指定します。この定義は，必ず tsdefvalue コマンドの -a オプションで指定してください。複数の環境変数を指定する場合は，次のようにします。

```
tsdefvalue /OTS completion_process_env -a "ENV1=VALUE1"¥
"ENV2=VALUE2" "ENV3=VALUE3"
```

指定できる定義の長さには制限があります。N 個の環境変数を指定する場合，次の条件を満たす必要があります。

（各文字列の長さの和）＋N＜＝ 2000

この定義にスペースが含まれる場合，スペース以降の環境変数は設定されません。次のように指定すると，「ENV3=VALUE3」は設定されません。

```
tsdefvalue /OTS completion_process_env -a "ENV1=VALUE1" ""
"ENV3=VALUE3"
```

決着プロセスホスト名　/OTS/completion_process_ipaddr_info

〜 < 文字列 ><<"">>

決着プロセスが使用するホスト名を明示的に指定します。IP アドレスは指定できません。マシンに複数のネットワークインタフェースがあり，そのうちの一つのネットワークインタフェースを使用する場合に，この定義を指定します。また，クラスタ構成でトランザクションサービスを運用する場合にも使用してください。この定

義が指定されていない場合は，デフォルトのホスト名が使用されます。再開始したときには，前回起動時に使用したホスト名を引き継ぎます。

例 `tsdefvalue /OTS completion_process_ipaddr_info  -s "host01"`

デーモンプロセスホスト名 /OTS/set_ipaddr_info

〜 < **文字列** ><<"">>

プライマリ IP アドレスが切り替わるクラスタ構成の場合に，OTS デーモン，決着デーモン，回復デーモン，および TCS の各プロセスが使用するホスト名を明示的に指定します。マシンに複数のネットワークインタフェースがあり，そのうちの一つのネットワークインタフェースを使用する場合に，この定義を指定します。この定義が指定されていない場合は，デフォルトのホスト名が使用されます。決着プロセスでは，/OTS/completion_process_ipaddr_info が指定されていれば，その指定が有効になります。

例 `tsdefvalue /OTS set_ipaddr_info  -s "host01"`

OTS **監視プロセス最大数**（C++） /OTS/max_process_monitor_count

〜 ((300 **〜** 30000))<<300>>

この定義は C++ 実行環境だけに提供される定義です。

OTS デーモンが監視できる C++ アプリケーションプロセスの最大数を指定します。OTS デーモンは，トランザクション実行中の C++ アプリケーションプロセスの状態を監視しています。これによって，トランザクション実行中の C++ アプリケーションプロセスが異常終了した場合，そのプロセスで実行中であったトランザクションのタイムアウトまで遅延することなく迅速に回復（ロールバック）処理が行われます。

この定義に指定する値の求め方を次に示します。

（OTS機能を使用するC++アプリケーションの最大同時存在プロセス数＋決着プロセス数＋回復プロセス数）＜（指定するOTS監視プロセス最大数）

TCS を使用する場合は，OTS 機能を使用するアプリケーションプログラムの最大同時存在プロセス数として 1 プロセスを追加してください。

なお，この定義の指定値が大きくなると，プロセス異常終了時の実行中のトランザクション回復までに時間が掛かることがあります。

トランザクションステータス書き込みモード /OTS/TM/set_status_write_mode "none"|"immediate"

〜 <<"none">>

トランザクションが完了したときのステータスを書き込むモードを指定します。次のどちらかを指定してください。

- "none"
  トランザクションが完了したとき，ステータスを書き込みません。通常運用時のレスポンス・スループットを向上できます。
- "immediate"
  トランザクションが完了したとき，トランザクション完了を示すステータスを書

き込みます。再起動時の OTS サービスの起動時間を短縮できます。

この定義が未設定の場合，メッセージ KFCB31493-W を出力し，"none" を使用します。
誤った値が指定されている場合，メッセージ KFCB31494-W を出力し，"none" を使用します。

トランザクション引き継ぎモード　/OTS/TM/set_recovery_mode 0|1

〜 <<0，1（Cosminexus TPBroker）>>

この定義は C++ 実行環境およびトランザクションコンテキストサーバに対して有効となる定義です。Cosminexus TPBroker では，必ず 1（デフォルト値）を指定してください。
リソースマネージャに対して発行した XA 関数がエラーリターンした場合のアプリケーションプログラムの処理を選択します。
決着処理失敗時の動作での，トランザクション決着処理モードの有効範囲を次の表に示します。

表 8-3　トランザクション決着処理モードの有効範囲

リターン値	XA 関数			
	xa_prepare	xa_commit	xa_rollback	xa_forget
XA_RETRY （連続して 5 回戻った場合）	−	○※	−	−
XA_RETRY_COMMFAIL （連続して 5 回戻った場合）	−	○※	○※	−
XAER_RMFAIL	○	○	○	○
XAER_RMERR	○	△	△	○

（凡例）
○：トランザクション決着処理モードが有効な範囲です。
−：XA 関数でリターン値が返されることはありません。
△：この定義の指定に関係なく，強制停止されません。

注※
XA 関数を一定回数リトライしても，該当リターン値が返り続けた場合。

- 0
  表 8-3 の○で示した部分で XA 関数がエラーリターンした場合，該当 XA 関数を発行したプロセスを強制停止します。
- 1
  表 8-3 の○で示した部分で XA 関数がエラーリターンした場合，該当 XA 関数を発行したプロセスを TPBroker が強制停止しないで，アプリケーションプログラムに制御を戻します。以降の決着処理は回復デーモンが引き継ぎます。すでにコ

ミット決定がなされていればコミットで，ヒューリスティック状態になっていれば対応するヒューリスティック例外で，それ以外の場合はロールバックでアプリケーションプログラムに制御を戻します。ただし xa_commit が 1 相コミットとして発行されていた場合は，HeuristicHazard 例外でアプリケーションプログラムに制御を戻します。

/OTS/TM/set_recovery_mode には，「1」を指定することをお勧めします。「1」を指定することによって得られる利点を次に示します。

- アプリケーションプログラムが不用意に終了してしまうケースを削減できます。
- マルチスレッドアプリケーションで，他スレッドの処理を巻き込んで終了することを回避できます。
- 強制終了（abort）が例外通知になることで，アプリケーションプログラム固有の後処理を行うことができます。

なお，この定義によるトランザクションの引き継ぎの指定は次の場合だけ有効です。

- すべてのリソースマネージャの /OTS/RM/set_xa_open_scope の設定が "thread"
- /OTS/RM/set_xa_rmerr_action の設定が "retry1"
- /OTS/RM/set_xa_rmfail_action の設定が "retry1"

この機能を有効にした場合，トランザクションが完了する前にアプリケーションプログラムへ制御を戻すために次のような事象が発生する場合があります。

- アプリケーションプログラムに制御を戻したあとに，ヒューリスティック状態になります。XA 関数からヒューリスティックのリターン値が返されたかどうかは，メッセージログおよびシステムログのメッセージ KFCB31200-E aa..aa function error. rmid=bb...bb rc=cc...cc（cc..cc が 5 ～ 8 の場合がヒューリスティックのリターン）から判断できます。
- xa_commit 関数が 1 相コミットとして発行されており，かつ回復デーモンが処理を引き継いだ場合に，アプリケーションプログラムへ HeuristicHazard 例外が返されます。

コンパイラ選択　/OTS/set_compiler_info "MS80"**（**Windows Visual Studio 2005**）の場合），**"SWCC"**（**Solaris **の場合），**"GLNX"|"GL32"**（**Linux AS4 **の場合）**

～ <<"SWCC"**（**Solaris **の場合），**"GLNX"**（**Linux AS4 **の場合）**>>

この定義は Windows（Visual Studio 2005）または Solaris または Redhat Enterprise Linux AS4 の C++ 実行環境だけに提供される定義です。TPBroker が使用するコンパイラを指定します。Windows（Visual Studio 2005）の場合 "MS80" を指定してください。Solaris の場合 "SWCC" を指定してください。RedHat Enterprise Linux AS4（x86）の場合 "GLNX" を指定してください。Redhat Enterprise Linux AS4（AMD64 & Intel EM64T）の場合 "GL32" を指定してください。AIX で Visual Age C++ for AIX V5.0 または V6.0 または IBM C/C++ Enterprise Edition for AIX V7.0 を使用する場合 "AIXC" を指定してください。AIX で IBM C/C++ Enterprise Edition for AIX V8.0 を使用する場合 "XLC8" を指定してください。

AIX で IBM C/C++ Enterprise Edition for AIX V9.0 を使用する場合 "XLC9" を指定してください。

tsstart コマンドタイムアウト値　/OTS/tsstart_timeout

〜 ((300 **〜** 1800))<<300>>(**単位：秒**)

tsstart コマンドで TPBroker を開始する場合のタイムアウト時間を指定します。

この定義が未設定の場合，メッセージ KFCB31493-W を出力し，タイムアウト時間には 300 が設定されます。誤った値が指定されている場合，メッセージ KFCB31494-W を出力し，タイムアウト時間には 300 が設定されます。

上記メッセージの出力先は，標準エラー出力，syslog，OTS メッセージログです。

なお，この定義は再開始後に引き継がれないで，常にシステム環境定義から読み込まれます。

例　`tsdefvalue /OTS tsstart_timeout -i 600`

8.3.3　リソースマネージャ定義（C++）

これらの定義は C++ 実行環境だけに提供されます。

リソースマネージャ定義は，Cosminexus TPBroker ではサポートしていません。

非同期インタバル　/OTS/RM/set_xa_async_interval

〜 ((1 **〜** 1000000))<<1000>>(**単位：ミリ秒**)

リソースマネージャに対して XA 関数を非同期で呼び出すときの，処理完了確認（xa_complete）発行インタバルを指定します。

オープン文字列　/OTS/RM/RMn/xa_open_string_info

〜 < **文字列** ><<"">>

TPBroker がリソースマネージャに対して xa_open 関数発行時に通知する文字列を指定します。文字列の内容は，使用するリソースマネージャの規定に従ってください。

クローズ文字列　/OTS/RM/RMn/xa_close_string_info

〜 < **文字列** ><<"">>

TPBroker がリソースマネージャに対して xa_close 関数発行時に通知する文字列を指定します。文字列の内容は，使用するリソースマネージャの規定に従ってください。

オープン文字列　/OTS/RM/RMn/DMN/xa_open_string_info

〜 < **文字列** ><<"">>

TPBroker のデーモンプロセスがリソースマネージャに対して xa_open 関数発行時に通知する文字列を指定します。文字列の内容は，使用するリソースマネージャの規定に従ってください。

クローズ文字列　/OTS/RM/RMn/DMN/xa_close_string_info

〜 < 文字列 ><<"">>

TPBroker のデーモンプロセスがリソースマネージャに対して xa_close 関数発行時に通知する文字列を指定します。文字列の内容は，使用するリソースマネージャの規定に従ってください。

オープン文字列　/OTS/RM/RMn/UAPn/xa_open_string_info

〜 < 文字列 ><<"">>

アプリケーションプログラムがリソースマネージャに対して xa_open 関数発行時に通知する文字列を指定します。文字列の内容は，使用するリソースマネージャの規定に従ってください。

クローズ文字列　/OTS/RM/RMn/UAPn/xa_close_string_info

〜 < 文字列 ><<"">>

アプリケーションプログラムがリソースマネージャに対して xa_close 関数発行時に通知する文字列を指定します。文字列の内容は，使用するリソースマネージャの規定に従ってください。

xa_open **発行タイミング**　/OTS/RM/set_xa_open_timing　"deferred" | "resolve"

〜 <<"deferred">>

リソースマネージャに対して xa_open 関数を発行するタイミングを指定します。次のどちらかを指定してください。

- "deferred"
  Current インタフェースの begin() の発行，TransactionFactory インタフェースの create() の発行，またはトランザクションのプロパゲーションによってトランザクションブランチが開始された時点で，リソースマネージャに xa_open 関数を発行します。
- "resolve"
  ORB_init() が発行された時点で，TPBroker に接続されている全リソースマネージャに xa_open 関数を発行します。

この定義は，リソースマネージャごとには指定できません。また，定義の変更は，変更後の TPBroker 開始時ではなく，変更後に起動されたプロセスから有効になります。回復処理への引き継ぎは行われません。xa_open および xa_close 関数の発行タイミングの詳細は，次の「xa_open 発行単位」を参照してください。

xa_open **発行単位**　/OTS/RM/set_xa_open_scope "process" | "thread"

〜 <<"process">>

TPBroker に接続されているリソースマネージャに対して xa_open および xa_close 関数を発行する単位を指定します。

/OTS/RM/RMn/set_xa_open_scope はリソースマネージャごとに指定できます。/OTS/RM/set_xa_open_scope は，/OTS/RM/RMn/set_xa_open_scope が設定されていない場合に有効になります。

- "process"

xa_open および xa_close 関数をそれぞれプロセスで 1 回ずつリソースマネージャに発行します。
/OTS/RM/set_xa_open_timing に "resolve" が設定されている場合は，プロセスで最初に ORB_init() が発行された時点で xa_open 関数を発行し，プロセスの停止時に xa_close 関数を発行します。ただし，クライアントからのコネクションで生成されたスレッドで ORB_init() を発行した場合は，クライアントとのコネクションに割り当てられた全スレッドが終了した時点で xa_close 関数が発行されます。
/OTS/RM/set_xa_open_timing に "deferred" が設定されている場合は，Current インタフェースの begin() の発行，TransactionFactory インタフェースの create() の発行，またはトランザクションのプロパゲーションのどれかがプロセスで最初に行われた時点で，リソースマネージャに xa_open 関数を発行します。トランザクショナルなクライアントでは，プロセスの停止時にリソースマネージャに xa_close 関数を発行します。サーバでは，オリジネータとのコネクションに割り当てられた全スレッドが終了した時点で，リソースマネージャに xa_close 関数が発行されます。

- "thread"
  xa_open および xa_close 関数をそれぞれスレッドで 1 回ずつリソースマネージャに発行します。
  /OTS/RM/set_xa_open_timing に "resolve" が設定されている場合は，ORB_init() の発行，Current インタフェースの begin() の発行，TransactionFactory インタフェースの create() の発行，またはトランザクションのプロパゲーションのどれかがスレッドで最初に行われた時点で，リソースマネージャに xa_open 関数を発行します。
  /OTS/RM/set_xa_open_timing に "deferred" が設定されている場合は，Current インタフェースの begin() の発行，TransactionFactory インタフェースの create() の発行，またはトランザクションのプロパゲーションのどれかがスレッドで最初に行われた時点で，リソースマネージャに xa_open 関数を発行します。
  xa_open 関数を発行したスレッドが終了するごとに，xa_close 関数が発行されます。

なお，/OTS/RM/RMn/set_xa_open_scope が設定されていない，かつ /OTS/RM/set_xa_open_scope が設定されている場合，/OTS/RM/set_xa_open_scope の設定値が有効になります。/OTS/RM/RMn/set_xa_open_scope および /OTS/RM/set_xa_open_scope が共に設定されていない場合は，デフォルト値の "process" になります。ただし，/OTS/RM/set_xa_open_scope をプロセスごとに定義することはできません。

リソースマネージャごとの定義

/OTS/RM/RMn のように定義キーを生成し，そのキーに対して xa_open_string_info および xa_close_string_info を指定します。RMn は，tsmkobj コマンドで指定したリソースマネージャ名称のことです。

プロセスごとの定義

/OTS/RM/RMn/ 名称のように定義を生成して行います。ここでは，回復プロセスでは DMN，アプリケーションプログラムプロセスおよび決着プロセスでは任意の名称を指定できます。各プロセスがどの定義キーを参照するかは，環境変数 TPRMINFO の値によって決まります。環境変数 TPRMINFO に値が設定されていない場合，および設定された値が示す定義キーが存在しないか，存在しても xa_open_string_info および xa_close_string_info が設定されていない場合は，/OTS/RM/RMn の定義キーを参照します。例えば，/OTS/RM/RM1/UAP1 のキーを参照させたいプロセスに対しては，環境変数 TPRMINFO に UAP1 を設定してください。

Oracle を使用する場合の定義

「8.4　定義例」を参照してください。

RMERR 動作モード　/OTS/RM/RMn/set_xa_rmerr_action "abort" | "retry1"

〜 <<"abort"**，**"retry1"**（**Cosminexus TPBroker**）**>>

リソースマネージャに対して発行した XA 関数が XAER_RMERR(-3) でリターンしたときの TPBroker の処理を選択します。この定義は，リソースマネージャごとに定義してください。また，各リソースマネージャに対して定義できるパラメタについては各リソースマネージャとの連携手順に従ってください。

- "abort"
  XA 関数がエラーリターンしたプロセスを異常終了させます。
- "retry1"
  即座に xa_close および xa_open 関数を発行後，エラーとなった XA 関数を再発行します。再発行した XA 関数の結果が再度エラーとなった場合は，そのままエラーは無視，xa_open 関数のエラーが発生した場合はプロセスを異常終了させます。また，このリトライは xa_start，xa_prepare，xa_forget 関数に対して行われます。

注

現在この定義によるリトライの指定は set_xa_open_scope が "thread" のリソースマネージャに対してだけ有効です。マルチスレッドライブラリ使用時に set_xa_open_scope に "thread" を指定可能なリソースマネージャには，"thread" を指定してください。なお，set_xa_open_scope に "process" を指定したリソースマネージャに対してこの定義のリトライ指定を行った場合は定義を無視して "abort" とみなします。

RMFAIL 動作モード　/OTS/RM/RMn/set_xa_rmfail_action "abort" | "retry1"

〜 <<"abort"**，**"retry1"**（**Cosminexus TPBroker**）**>>

リソースマネージャに対して発行した XA 関数が XAER_RMFAIL(-7) でリターンしたときの TPBroker の処理を選択します。この定義は，リソースマネージャごとに定義してください。また，各リソースマネージャに対して定義可能なパラメタについては各リソースマネージャとの連携手順に従ってください。

- "abort"
  XA 関数がエラーリターンしたプロセスを異常終了させます。
- "retry1"
  即座に xa_open 関数を発行後，エラーとなった XA 関数を再発行します。再発行した XA 関数の結果が再度エラーとなった場合は，そのままエラーとして処理を続行します。ただし，処理が続行できない場合はプロセスを異常終了させることがあります。xa_close および xa_open 関数の発行処理では xa_close 関数のエラーは無視，xa_open 関数のエラーが発生した場合はプロセスを異常終了させます。
  また，このリトライは xa_start，xa_prepare，xa_commit，xa_rollback，xa_forget 関数に対して行われます。

注

現在この定義によるリトライの指定は set_xa_open_scope が "thread" のリソースマネージャに対してだけ有効です。マルチスレッドライブラリ使用時に set_xa_open_scope に "thread" を指定可能なリソースマネージャには，"thread" を指定してください。なお，set_xa_open_scope に "process" を指定したリソースマネージャに対してこの定義のリトライ指定を行った場合は定義を無視して "abort" とみなします。

8.3.4　回復定義

回復インタバル　/OTS/RCV/set_retry_time

〜 ((1 **〜** 65535))<<10>>(**単位：秒**)

回復する必要のあるトランザクションブランチの検索，および回復処理を行う間隔を秒単位で指定します。
アプリケーションプログラムが異常終了したときに何らかの原因でトランザクションブランチの回復処理ができなかった場合，そのトランザクションブランチの回復はここで指定したインタバルごとに試みられます。この間隔を短くすると，システムのスループットが悪くなりますので注意してください。

例　`tsdefvalue /OTS/RCV set_retry_time -i 10`

トランザクション回復タイミング　/OTS/RCV/set_startup_recovery_skip 0|1

1 **〜** <<0，1（05-18 **以降，および** Cosmienxus TPBroker）>>

＜ 05-18 より前のバージョンの場合＞

再開始モードの場合，トランザクション回復処理を，tsstart 実行中 /tsstart 終了後のどちらで行うかを設定します。値を 0 に設定した場合，tsstart 実行中にトランザクション回復処理を実行します。値を 1 に設定した場合，tsstart 終了後に，初回 0 秒後（tsstart 完了直前に開始），2 回目以降は /OTS/RCV/set_retry_time で指定された間隔でトランザクション回復処理を実行します。値を 0 に設定した場合，リソースマネージャが未起動状態など，回復ができない場合，tsstart コマンドがタイムアウトを起こす場合があります。値を 1 に設定した場合，トランザクションを管理するステータスファイルが，未決着トランザクションの情報で一杯の場合に，

tsstart コマンドがリターンした直後にトランザクションを開始できない場合があります。この場合，KFCB31222-E メッセージが出力されます。この未決着トランザクションの情報が一杯の状態は，主にシステム定義 /OTS/TM/set_status_write_mode を "none" に設定している場合に発生します。
＜ 05-18 以降の場合＞
再開始モードの場合に，トランザクション回復処理方法を設定します。指定された値が 0 または 1 以外の値を設定した場合はデフォルト値 "1" として動作いたします。

- 0
  tsstart コマンド実行中にトランザクション回復処理を実行します。リソースマネージャの未起動状態などが原因で回復できないトランザクションがある場合，/OTS/RM/set_recovery_retry_count に指定された回数リトライを行い OTS の起動に失敗する場合があります。
- 1
  tsstart コマンド実行中のトランザクション回復処理に失敗した場合でも，tsstart コマンドは成功します。トランザクション回復処理は，tsstart コマンド終了後に行います。初回は 0 秒後（tsstart コマンド終了直後）に，2 回目以降は回復定義 /OTS/RCV/set_retry_time に指定された時間が経過した後に，トランザクション回復処理を実行します。以下の状態の場合，トランザクションの回復中であるため，新しいトランザクションを開始出来ない場合があります。回復処理中にエラーが起きている可能性があります。KFCB31222-E または KFCB31226-E 以外のメッセージが出力されていないか確認してください。エラーが発生している場合は，エラーの原因を解決してから新しいトランザクションを開始してください。

- トランザクションを管理するステータスファイルが，未決着トランザクションの情報で空きがなくなり，新しいトランザクションの情報を書き込めない場合
  この場合，KFCB31222-E メッセージが出力されます。ステータスファイルが未決着トランザクションの情報で空きがなくなる状態は，主にトランザクション定義 /OTS/TM/set_status_write_mode に ” none ” を指定している場合に発生します。
- 回復処理中（リソースマネージャへ xa_recover() を発行中）の場合
  この場合，KFCB31226-E メッセージが出力されます。xa_recover() が失敗する原因として，主にリソースマネージャが未起動状態であることが考えられます。新規トランザクションは，KFCB31512-I メッセージが出力されると開始することができます。tsstart コマンド終了後の回復処理の場合は，システム +B196 定義 /OTS/RCV/set_recover_retry_count は無効になり，決着するまで処理を繰り返し行います。

8.3.5 トランザクションコンテキストサーバ定義（Java）

これらの定義は Java 実行環境だけに提供されます。

トランザクションコンテキストサーバ定義は，Cosminexus TPBroker では有効になりま

せん。

TCS 名　/OTS/TCS/trn_ctx_sv_name

～ < 文字列 >((0 ～ 16 文字))<<"">>

TCS の名前を指定します。

Java トランザクショナルアプリケーションが，特定の TCS と接続するには，TpCosOTS.set_property で指定した TCS 名と，この定義で指定した TCS 名を一致させる必要があります。最大で 16 文字の文字列を指定できます。16 文字を超えた名前を指定した場合，org.omg.CORBA.BAD_PARAM 例外が発生します。

また，"　"のようにすべてスペースでも指定できません。名前の両端にスペースを含んでいる場合，これらのスペースは削除されます。TpCosOTS.set_property については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

この定義で TCS 名が指定されていても，名前を指定していない Java トランザクショナルアプリケーションからの接続は受け付けます。また，この定義が未設定の場合，すべての Java トランザクショナルアプリケーションからの接続を受け付けます。

例　`tsdefvalue /OTS/TCS trn_ctx_sv_name -s "sample"`

トランザクションデフォルトタイムアウト値　/OTS/TCS/transaction_default_timeout

～ ((0 ～ 2147483647))<<180>>(単位：秒)

トランザクションがタイムアウトするまでの時間（秒）を指定します。Java トランザクショナルアプリケーションが，トランザクションのタイムアウト値を 0 に設定したときに適用するタイムアウト値を指定します。この定義に 0 を指定している場合，Java トランザクショナルアプリケーションがタイムアウト値を 0 に設定したトランザクションではタイムアウトを起こしません。

例　`tsdefvalue /OTS/TCS transaction_default_timeout -i 180`

8.3.6 システム定義

システム識別子情報　/SYSTEM/system_id_info

～ < 文字列 >((4 文字))<<"">>

システム識別子情報を 4 文字の文字列で指定します。

この定義が未設定の場合，および空白文字列が指定された場合は，ホスト名から自動的に識別子が生成されます。指定値が 4 文字を超える場合は，先頭の 4 文字が設定されます。4 文字に満たない場合は，不足部分が「_」で補足されます。文字列の途中にスペースがある場合は，スペースが「_」に置き換えられます。

この定義は，同一 ORB ドメイン内で複数の OTS 実行環境を実現するときに自動生成された識別子が衝突する場合やクラスタ構成で OTS 機能を使用するときにシステム識別子を統一する必要がある場合に指定します。

8.3.7 トランザクショントレース定義

トランザクションブランチの決着状況　/OTS/TRC/set_trntrace_level　"err" | "all" | "none"

〜 <<"err">>

どのような場合にトランザクショントレースをトレースファイルに記録するかを指定します。

- "err"
  トランザクションブランチの決着状況がロールバックとヒューリスティック決着の場合だけ記録します。
- "all"
  すべてのトランザクションブランチ情報を記録します。
- "none"
  トランザクショントレースを記録しません。

TPBroker は，トランザクションブランチが定義で指定された状態で決着した場合だけ，トレースファイルに記録します。ヒューリスティック決着の詳細については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

トレースファイルサイズ　/OTS/TRC/max_trntrace_file_size

〜 ((1024 〜 1048576))<<1024>>(**単位：キロバイト**)

トランザクショントレースファイルの最大容量を指定します。

指定する単位はキロバイトです。ファイル容量を 1 メガバイトとする場合，1024 と指定します。ファイルサイズがこの値を超過するとトレースファイルが新規に作成され，元のトランザクショントレースファイルはバックアップとして残されます。

トレースファイル世代数　/OTS/TRC/max_trntrace_file_count

〜 ((3 〜 256))<<3>>

出力するトランザクショントレースファイルの世代数を指定します。トレースファイル名は trn_xxx（xxx：001 〜定義で指定した値）です。

トレース情報がファイルサイズを超過すると，次のファイルに切り替わり運用を続けます。トレースファイルの運用の詳細については，「7.7.1　トレースファイル」を参照してください。

8.4 定義例

ここでは，TPBroker の定義例を示します。

8.4.1 リソースマネージャとの XA 連携（C++）

リソースマネージャと XA 連携を行う場合には，X/Open の XA インタフェースで定められているリソースマネージャのオープン，クローズ文字列の設定を定義で指定する必要があります。このときに，tskeycreate，tsdefvalue コマンドを使用します。リソースマネージャとして Oracle を使用する場合の例を，次に示します。各リソースマネージャのオープン，クローズ文字列に指定する内容については，リソースマネージャのバージョンによって異なる場合があるため，それぞれの製品のマニュアルを参照してください。

リソースマネージャとの XA 連携は，Cosminexus TPBroker ではサポートしていません。

● Oracle の場合

```
#for all UAP(default)
tskeycreate /OTS/RM/Oracle_XA
tsdefvalue /OTS/RM/Oracle_XA ¥
xa_open_string_info ¥
   -s "Oracle_XA+Acc=P/demo/demo+SesTm=15"
tsdefvalue /OTS/RM/Oracle_XA ¥
xa_close_string_info ¥
   -s "Oracle_XA+Acc=P/demo/demo+SesTm=15"

#for Daemon
tskeycreate /OTS/RM/Oracle_XA/DMN
tsdefvalue /OTS/RM/Oracle_XA/DMN ¥
   xa_open_string_info -s ¥
   "Oracle_XA+Acc=P/sys/oracle+SesTm=15"
tsdefvalue /OTS/RM/Oracle_XA/DMN ¥
   xa_close_string_info -s ¥
   "Oracle_XA+Acc=P/sys/oracle+SesTm=15"
```

8.4.2 リソースマネージャの削除（C++）

登録していたリソースマネージャを削除する場合には，同時にリソースマネージャのオープン，クローズ文字列の定義も削除します。このときに，tsdefremove コマンドを使用します。リソースマネージャとして Oracle を使用する場合の定義を削除する例を次に示します。

リソースマネージャの削除は，Cosminexus TPBroker ではサポートしていません。

● Oracle の場合

```
tsdefremove /OTS/RM/Oracle_XA/DMN xa_open_string_info
tsdefremove /OTS/RM/Oracle_XA/DMN xa_close_string_info
```

```
tsdefremove /OTS/RM/Oracle_XA xa_open_string_info
tsdefremove /OTS/RM/Oracle_XA xa_close_string_info

tskeyremove /OTS/RM/Oracle_XA/DMN
tskeyremove /OTS/RM/Oracle_XA
```

8.4.3 プロセス監視定義の定義例

プロセス監視定義の定義例を次に示します。次に示すプロセス監視定義は，$TPDIR/adm/admconf.cf にあります。

```
# TPBrokerプロセス監視定義
#           サンプル用定義ファイル
#OSAgent
OSAG:"/opt/TPBrokerV5/bin/osagent -g": : : ¥
1:restart:down: :none:3: : : : :

#OTS-Daemon
0001:"/opt/TPBrokerV5/bin/tsstart":¥
"/opt/TPBrokerV5/bin/tsotspid": : ¥
600:none:restart: :none:3:¥
"/opt/TPBrokerV5/bin/tsstop":¥
"/opt/TPBrokerV5/bin/tsstop -f2":¥
 : :
```

9 運用コマンド

この章では，TPBroker で使用する運用コマンドの概要と詳細について説明します。

9.1　運用コマンドの概要

ここでは，各運用コマンドに共通する内容として，運用コマンドの入力方法および記述形式について説明します。

9.1.1　運用コマンドの入力方法

TPBroker の運用コマンドは，シェルから入力してください。

9.1.2　運用コマンドの記述形式

運用コマンドの記述形式を次に示します。

```
コマンド名　-オプションフラグ〔フラグ引数〕　コマンド引数
```

(1)　コマンド名

コマンド名は，実行するコマンドのファイル名です。

TPBroker の運用コマンドは $TPDIR/bin/ にあるので，環境変数 PATH に $TPDIR/bin（Windows の場合は %TPDIR%¥bin）を設定してください。

(2)　- オプションフラグ〔フラグ引数〕

オプションはマイナス記号（-）で始まる文字列で，フラグ引数を取らないか，または 1 個のフラグ引数を取ります。

オプションフラグ

1 文字の英数字（大文字と小文字は区別されます）

フラグ引数

オプションフラグに対する引数

注意

- フラグ引数を必要とするオプションフラグのフラグ引数は省略できません。
- オプションはコマンド引数よりも前に指定しなければなりません。
- マイナス記号だけのオプションは入力できません。
  例　次のように入力すると，- はコマンド引数とみなされます。

```
prompt> cmd -
```

cmd はコマンド名称です。

(3)　コマンド引数

コマンド引数は，コマンド操作の対象となるものを指定します。

9.2　TPBroker で使用する運用コマンド

TPBroker で使用する運用コマンドの一覧を次の表に示します。詳細は，「9.3　運用コマンドの詳細」で説明します。VisiBroker の運用コマンドについては，マニュアル「Borland(R) Enterprise Server VisiBroker(R)　プログラマーズリファレンス」を参照してください。

表 9-1　運用コマンドの一覧

コマンドの種類	コマンド名
運用支援機能の運用コマンド	admexec（vbj コマンド，ネーミングサービスの監視）（Java）（Windows XP）※1
	admlaunch（vbj コマンド，ネーミングサービスの監視）（Java）（Windows XP，Windows 2003）※1
	admlogcat（メッセージログの出力）
	admlsenv（環境変数の情報の出力）※1
	admlsprc（監視対象プロセスの情報の表示）
	admreload（プロセス監視定義ファイルの再読み込み）
	admsetup（実行環境のセットアップ）
	admstart（TPBroker の開始）※2
	admstartprc（プロセスの起動と監視の開始）※2
	admstat（TPBroker の稼働情報表示）
	admstop（TPBroker の終了）※2
	admstopprc（監視の終了とプロセスの停止）※2
OTS の運用コマンド	trnctxsv（トランザクションコンテキストサーバの起動）（Java）※3
	tscommit（トランザクションのコミット）
	tsedapt（API トレースファイル解析）（C++）※4
	tsedtrntrc（トランザクショントレースの出力）※5
	tslnkrm（リソースマネージャの登録・削除）（C++）※4
	tslogcat（メッセージログの出力）
	tslsfs（TPBroker ファイルシステムの内容表示）※5
	tslsrm（リソースマネージャ情報の表示）（C++）※4
	tslstrn（トランザクションの状態表示）
	tsmkfs（TPBroker ファイルシステムの初期設定）※5

コマンドの種類	コマンド名
	tsmkobj（トランザクション制御用オブジェクトファイルの作成）（C++）※4
	tsrollback（トランザクションのロールバック）
	tssetup（TPBroker のセットアップ）
	tsstart（トランザクションサービスの開始）※3
	tsstat（OTS の状態表示）
	tsstatfs（TPBroker ファイルシステムの状態表示）※5
	tsstop（トランザクションサービスの終了）※3
	tsstoptrnctxsv（トランザクションコンテキストサーバの終了）（Java）※3
	tstrnsts（トランザクション稼働統計情報の出力）
TPBroker の定義設定コマンド	tsdefremove（定義パラメタの削除）
	tsdefvalue（定義パラメタへの指定値の設定）
	tskeycreate（定義キーの生成）
	tskeyremove（定義キーの削除）
	tslsconf（定義パラメタの表示）
TPBroker の障害情報取得コマンド	tsrasget（障害調査資料の採取）※5

注※1　Windows 版だけに提供されるコマンドです。

注※2　tsstart コマンドで開始した OTS 環境では使用できません。

注※3　admstart コマンドで開始した TPBroker 環境では使用できません。

注※4　Cosminexus TPBroker ではサポートしていません。

注※5　UNIX 版だけに提供されるコマンドです。

9.3　運用コマンドの詳細

表 9-1 に示した運用コマンドの詳細をアルファベット順に説明します。

admexec（vbj コマンド，ネーミングサービスの監視（Java）（Windows XP））

形式，機能，オプション，記述例，注意事項については，admlaunch コマンドと同一です。admlaunch コマンドを admexec に読み替えてください。

admlaunch（vbj コマンド，ネーミングサービスの監視（Java）（Windows XP，Windows 2003））

● admlaunch コマンドは，Windows Vista 対応版以降 admexec に名称を変更しました。

形式

admlaunch -i 識別子 [起動コマンド列ファイル]

機能

Windows 版で TPBroker の運用支援機能を使用して vbj コマンド（起動する Java アプリケーションを含む）およびネーミングサービス（nameserv プロセス）を監視するための，起動用コマンドおよび停止用コマンドです。プロセス監視定義中または admstartprc，admstopprc コマンドの引数として使用します。

admlaunch コマンドで起動した監視対象プロセスは，必ず停止用コマンドとして動作する admlaunch コマンドを使用して停止させてください。

なお，これは Windows 版だけに提供されるコマンドです。

オプション

-i 識別子

直接起動方式のプロセスとして運用監視機能に登録する際に指定した識別子と同じ識別子を指定します。識別子の指定値については，「8.2　プロセス監視定義の詳細」を参照してください。

引数

起動コマンド列ファイル

実際に起動する vbj コマンドまたはネーミングサービス（nameserv プロセス）のコマンド列を格納したテキストファイルを用意し，そのファイル名をフルパスで指定します。スペースを含むパスを指定する場合は，パス全体を「¥"…¥"」のように「¥"」で囲んでください。このファイル名には環境変数は記述できません。

このオプションは，admlaunch コマンドを起動用コマンドとして使用するときだけ指定します。指定しないときは，admlaunch コマンドは停止用コマンドとして動作します。

用意するテキストファイルには，admlaunch コマンドで起動するコマンド列を一行に記述します。一行の長さは 2047 文字以内です。テキストファイルは，最初の一行しか読まれません。

コマンド列の第 1 引数は，vbj コマンドまたはネーミングサービス（nameserv プロセス）への絶対パスを指定します。また，ログオフ時に終了しないオプションを指定します。スペースを含むパスを指定する場合は，パス全体を「"…"」のように「"」で囲んで

ください。

記述例

プロセス監視定義の記述例を次に示します。

記述例 1

```
JSVA※1:"C:¥TPBrokerV5¥bin¥admlaunch.exe -i JSVA ※2
C:¥TPBrokerV5¥java examples¥ots¥server2.txt": ¥※3
: : :none:restart: :none:3 ¥
:"C:¥TPBrokerV5¥bin¥admlaunch.exe -i JSVA": : : ¥※4
:"CLASSPATH=C:¥TPBrokerV5¥examples¥ots;C:¥TPBrokerV5¥lib¥vbjorb.ja
r"※5
```

記述例 2

```
JSVB:"¥"C:¥Program Files¥TPB¥bin¥admlaunch.exe¥"※6 -i JSVB
¥"C:¥Program Files¥Sample¥client.txt¥"※6": ¥
: : :none:restart: :none:3 ¥
:"¥"C:¥Program Files¥TPB¥bin¥admlaunch.exe¥"※6 -i JSVB": : : ¥
:"CLASSPATH=C:¥Program Files¥Sample;C:¥Program
Files¥TPB¥lib¥vbjorb.jar※7"
```

注※ 1

admlaunch コマンドは直接起動方式のプロセスです。この場合，識別子は「JSVA」になります。

注※ 2

起動用コマンドの admlaunch コマンドの引数に，-i オプションで注※ 1 の識別子を指定してください。

注※ 3

起動するプロセスのコマンド列を記述したファイルを用意し，そのファイルの絶対パスを指定します。

注※ 4

停止用コマンドも admlaunch コマンドです。-i オプションで注※ 1 の識別子を指定します。それ以外のオプションは指定しません。

注※ 5

プロセス起動時に設定が必要な環境変数名と値は，プロセス監視定義の最後に記述します。

注※ 6

スペースを含むパスを指定する場合は，パス全体を「¥"…¥"」のように「¥"」で囲んでください。

注※ 7

環境変数に指定するパスは，「¥"…¥"」のように囲みません。

起動コマンド列ファイルの内容例を次に示します。

内容例 1

```
C:¥TPBrokerV5¥bin¥vbj.exe※1 -J-Xrs※2
-Dorg.omg.PortableInterceptor.ORBInitializerClass.COM.Hitachi.soft
ware.TPBroker.OTS.Init Server2Main
```

内容例 2

```
"C:¥Program Files¥TPB¥bin¥vbj.exe"※3 -J-Xrs SampleClient
```

注※ 1

監視対象の vbj.exe コマンドを絶対パスで記述します。

注※ 2

ログオフ時に終了しないオプションを指定します。ログオフ時に終了しないオプションについては，Java VM の Readme などを参照してください。

注※ 3

スペースを含むパスを指定する場合は，パス全体を「"…"」のように「"」で囲んでください。

注意事項

- admlaunch コマンドで正しく監視できるのは，vbj コマンドおよびネーミングサービスだけです。ほかのプロセスを指定した場合，動作は保証されません。
- 識別子を誤った場合は，プロセスが正常に起動または停止しない場合があります。その場合は，メッセージに従って識別子を確認してください。
- タスクマネージャで admlaunch.exe コマンドプロセス，vbj.exe コマンド，nameserv プロセスなどを直接停止させないでください。

admlogcat（メッセージログの出力）

形式

admlogcat [メッセージログファイル名]

機能

TPBroker の運用支援機能が出力したメッセージを標準出力に出力します。

引数

メッセージログファイル名

ADM デーモンが出力したメッセージログファイルを指定します。過去のメッセージログファイルは，TPBroker が作成したバックアップ用の $ADMSPOOL+_logN 下の log/admlog です。メッセージログファイル名を指定しない場合は，デフォルトとして $ADMSPOOL/log/admlog に出力されたメッセージの内容が出力されます。TPBroker 稼働中は，稼働中のシステムが出力しているメッセージの内容が出力されます。

指定したファイルが，ADM デーモンが出力したメッセージログファイル以外の場合は，何も出力されないでコマンドが終了します。

表示形式

```
1999/09/28 21:23:59 169 207 KFCB29125-I ADMD started (PID=169,
TIME=Tue Sep 28 21:23:58 1999).

1999/09/28 21:23:59 169 207 KFCB29132-I ADMD started manually.

1999/09/28 21:24:14 169 207 KFCB29128-I admstart is executed.

1999/09/28 21:24:14 169 207 KFCB29133-I ADMD started with normal
mode.

1999/09/28 21:24:23 169 207 KFCB29128-I admstop is executed.

1999/09/28 21:24:23 169 207 KFCB29136-I ADMD stopped with normal
mode.
```

admlsenv（環境変数の情報の出力（Windows））

形式

admlsenv [-r サービス名 | -a]

機能

サービスに登録されている TPBroker 環境の環境変数の情報を出力します。オプションが指定されていない場合，「TPBroker」としてサービスに登録されている TPBroker 環境の環境変数の情報を出力します。

なお，これは Windows 版だけに提供されるコマンドです。

オプション

-r サービス名

TPBroker のサービス名称を指定します。このオプションが設定されていると，admlsenv は指定したサービス名で登録した TPBroker 運用支援機能の実行環境に定義された環境変数の情報，および TPBroker のバージョンとプロセス監視定義ファイルのフルパスを出力します。

-a

-a オプションはサービスとして登録されているすべて TPBroker 運用支援機能の実行環境を対象に，環境変数の情報，および TPBroker のバージョンと定義ファイルのパスを表示する場合に指定します。

表示形式

```
prompt> admlsenv -a
[TPBroker2]
Version  = 05-15
TPDIR    = C:¥TPBroker
TPSPOOL  = D:¥work¥SPOOLS¥otsspool
TPFS     = D:¥work¥SPOOLS¥otsspool
ADMSPOOL = D:¥work¥SPOOLS¥admspool
ADMFS    = D:¥work¥SPOOLS¥admfs
AdmConfFile = "C:¥TPBroker¥adm¥admconf.cf"
[TPBrokerDevelop]
Version  = 05-15
TPDIR    = C:¥TPBroker
TPSPOOL  = C:¥TPBroker¥otsspool
TPFS     = C:¥TPBroker¥otsspool
ADMSPOOL = C:¥TPBroker¥spool
ADMFS    = C:¥TPBroker¥spool
AdmConfFile = "C:¥TPBroker¥adm¥admconf.cf"
```

注意事項

admsetup コマンドが実行されてない状態で admlsenv コマンドを実行すると，エラーが

発生します。

admlsprc（監視対象プロセスの情報の表示）

形式

admlsprc [-l]

機能

ADM デーモンが監視しているプロセスについての情報を表示します。

プロセスに関する情報を次に示します。

識別子
監視中のプロセスを識別するための識別子です。プロセス監視定義ファイルに設定したプロセスの場合は，プロセス監視定義ファイル中に指定された識別子を指定します。動的にプロセス監視に参加したプロセスの場合は任意に割り当てられます。

パス名
現在監視中のプロセスのパス名またはプロセス名です。

プロセス ID
現在監視中のプロセスのプロセス ID です。

監視種別
プロセス監視定義ファイルに設定されたプロセスか，動的に参加したプロセスかを表します。

- CONF_A
  プロセス監視定義ファイルに設定されたプロセス（直接起動方式）
- CONF_B
  プロセス監視定義ファイルに設定されたプロセス（間接起動方式）
- DYNA_A
  API を使用して動的にプロセス監視に参加したプロセス
- DYNA_B
  admstartprc コマンドを使用して動的にプロセス監視に参加したプロセス

監視時間
監視開始からこのコマンド入力時までの監視時間です。「時：分：秒」の形式で表示されます。

オプション

-l

上記に示したプロセス情報がすべて表示されます。このオプションを省略した場合は，識別子だけが表示されます。

表示形式

```
prompt> admlsprc -l
ID       FILE                           PID    KIND     TIME
--------------------------------------------------------------
OSAG     /TPBroker/bin/osagent.exe -c   122    CONF_A   1:01:40
0001     otsd                           123    CONF_B   1:01:40
TOTAL 2

prompt> admlsprc
OSAG  0001
TOTAL 2
```

注意事項

- このコマンドは，ADM のデーモンがコマンド受信可能状態の場合に有効です。プロセス監視定義ファイルの設定によって，デーモンプロセスがコマンド受信可能状態になるまでに時間が掛かる場合があります。
- ADM デーモンが稼働していない場合は，このコマンドはメッセージを出力して終了します。
- UNIX 版の場合，間接起動方式で起動または監視したプロセスのプロセス名をこのコマンドで表示させると，プロセス名の長さが実際のプロセスよりも短く表示されます。HP-UX では最大 14 文字，Solaris では最大 15 文字，AIX では最大 32 文字，Linux では最大 15 文字です。

admreload（プロセス監視定義ファイルの再読み込み）

形式

admreload { -i 識別子 [識別子]... | -a } [-f] [-l]

機能

プロセス監視定義ファイルを再読み込みします。

オプション

-i 識別子

プロセス監視定義ファイル上の識別子で指定された定義を再読み込みの対象とします。

-a

プロセス監視定義ファイル全体を再読み込みの対象とします。

-f

定義に変更，削除があり，かつ該当プロセスが起動中であった場合，起動中のプロセスを停止させ，変更，削除を反映します。

このオプションが指定されなかった場合，起動中のプロセスの変更，削除は反映しません。

-l

定義の変更の場合，変更状況の詳細を出力します。

表示形式

```
識別子：変更タイプ（added，changedまたはdeleted）
項目（詳細は表9-2を参照のこと）：
before："変更前の内容"（定義なしの場合"-"）
after ："変更後の内容"（定義なしの場合"-"）
項目：～（以下，項目，before，afterの繰り返し）
識別子：～（以下，識別子の繰り返し）
```

admreload コマンドで表示される項目を次の表に示します。

表 9-2　項目一覧

定義フィールド通番	項目	説明
1	ORDER	起動順序
2	NORMAL-PATH	正常起動パス
3	RECV-PATH（直接起動の場合）	再起動パス

定義フィールド通番	項目	説明
	GETPID-PATH （間接起動の場合）	PID 取得コマンドパス
4	FORCE-PATH	強制正常起動パス
5	WAIT-TIME	起動用コマンドタイムアウト値
	STOP-WAIT-TIME	停止用コマンドタイムアウト値
6	INVOKE-FAIL	プロセス起動失敗アクション
7	MONITOR-FAIL	プロセス異常終了時アクション
8	MON-FAIL-PATH	異常終了時実行パス
9	WHEN-INVOKE	プロセス起動タイミング
10	DOWN-COUNT	連続異常終了回数
	DOWN-RETRY-INTERVAL	監視時間 （HP-UX(IPF) 05-16-/B 以降，それ以外のプラットフォームは 05-17 以降のみ）
11	STOP-PATH	正常停止用コマンド
12	F-STOP-PATH	強制停止用コマンド
13	UID	コマンド実行ユーザ ID
14	GID	コマンド実行グループ ID
15	ENVIRONMENT	プロセスごとの環境変数

admsetup（実行環境のセットアップ）

形式

admsetup { -d [-r サービス名] [-f] ｜ -c プロセス監視定義ファイル名 [-r サービス名] [-i] [-n]}

機能

TPBroker の実行環境を初期化します。また，作業用のディレクトリの作成，初期化，および OS への登録を行います。

UNIX 版の場合，コマンドを実行した環境に設定されている次の環境変数を運用支援機能の開始時に引き継ぎます。

ADMFS，ADMSPOOL，LANG，LD_LIBRARY_PATH（Solaris，Linux の場合），LIBPATH（AIX の場合），OSAGENT_PORT，SHLIB_PATH（HP-UX の場合），TPDIR，TPFS，TPRMINFO，TPSPOOL，TZ，VBROKER_ADM

Windows 版の場合，運用支援機能の実行環境をサービスとして OS に登録し，admsetup コマンドを実行した TPBroker 環境の環境変数を設定します。admsetup コマンドを実行した環境に設定されている，次の環境変数を運用支援機能の開始時に引き継ぎます。

ADMFS，ADMSPOOL，TPDIR，TPFS，TPSPOOL

オプション

-d

実行環境が削除され，OS から登録解除されます。

-f

クラスタシステム構成で環境変数 ADMFS を共有している場合，および TPBroker 運用支援機能の引き継ぎ情報を削除する場合に指定します。このオプションを指定して admsetup コマンドを実行した場合，環境再構築後最初の運用支援機能の起動は正常開始となります。

-f オプションは，必ず -d オプションとともに指定してください。

-c **プロセス監視定義ファイル名**

プロセス監視定義を設定しているプロセス監視定義ファイル名を，ドライブ名を含む絶対パスで指定します。プロセス監視定義ファイルの記述形式については「8.2　プロセス監視定義の詳細」を参照してください。

-i

環境変数 ADMFS で設定されたディレクトリ，またはキャラクタ型スペシャルファイルを，複数の TPBroker 環境で共有する場合に指定します。このオプションを指定した場合と指定しない場合の動作を次の表に示します。

表 9-3　-i オプションを指定する場合と指定しない場合の動作（admsetup コマンドの場合）

動作	-i オプションあり	-i オプションなし
$ADMFS ディレクトリがある場合の admsetup コマンドの動作	正常終了	異常終了
$ADMFS ディレクトリの作成タイミング	admstart コマンド実行時	admsetup コマンド実行時
-d オプションを指定して admsetup コマンドを実行した場合に $ADMFS ディレクトリを削除するかどうか	削除しない	削除する

-i オプションを指定すると $ADMFS ディレクトリは削除されないため，どの環境からも参照されなくなった $ADMFS ディレクトリは OS のコマンドなどで削除してください。また，-d オプションを指定して admsetup コマンドを実行すると，$ADMSPOOL ディレクトリは削除されるため，-d オプションを指定して admsetup コマンドを実行したあとに，-i オプションを指定して admsetup コマンドを実行する場合は，環境変数 ADMFS に環境変数 ADMSPOOL とは別のパスを明示的に設定してください。

ただし，同一ノードで複数の TPBroker を OS に登録する場合は，このオプションを指定しないでください。

-n

admsetup コマンド実行後の最初の ADM の起動を正常開始にします。$ADMFS ディレクトリを共用ディスクや TPBroker ファイルシステムに配置すると，監視対象プロセスの情報が前回の ADM 停止時の状態で残る場合があります。このオプションは，ADM 環境を作成し直す場合で監視対象プロセスの情報を引き継ぐ必要がないときに指定してください。

-r サービス名

登録または削除するサービス名を指定します。サービス名は，(空白と「_」(アンダースコア) を含む半角英数字を 1 ～ 32 文字で指定します。大文字小文字は区別しません。また，サービス名の先頭および語尾に空白文字は指定できません。このオプションを省略した場合，「TPBroker」を名称としてサービスを登録または削除します。

このオプションは，Windows 版 TPBroker 05-15 以降で有効です。

表示形式

```
prompt> admsetup -c $TPDIR/adm/admconf.cf
KFCB29031-I admsetup successful.

prompt> admsetup -d
KFCB29031-I admsetup successful.
```

注意事項

- -d オプションを指定した場合，作業用ディレクトリ $ADMSPOOL および $ADMFS が削除されます。TPBroker の稼働中は，-d オプション付きの admsetup コマンドは実行しないでください。
- このコマンドを実行できるのは，UNIX 版では root 権限，Windows 版では Administrator 権限を持つユーザです。
- -f オプションが -d オプションとともに指定されていない場合，メッセージ KFCB29012-I を出力して削除処理は行いません。
- ファイルの削除に失敗した場合でも，削除処理は続行して削除できるファイルをすべて削除します。その場合，処理中の下位レイヤでエラーが発生したとき，メッセージ KFCB29073-W を出力します。メッセージの詳細は，「11.2　メッセージ一覧」を参照してください。
- admsetup コマンドで OS に登録できる TPBroker 運用支援機能の実行環境の数は，UNIX 版は 5 個，Windows 版は制限がありません。
- UNIX 版の場合，運用支援機能の実行環境を /etc/inittab に登録するときに TPBroker が使用する識別子は，HP-UX では tp00 〜 tp09，Solaris では g0 〜 g9，AIX および Linux では t0 〜 t9 です。
- OTS 機能を使用しないで TPBroker の運用支援機能だけを使用する場合でも，admsetup コマンド実行前に tssetup コマンドを実行してください。
- admsetup コマンドで TPBroker を OS へ登録した場合，TPBroker をアンインストールする前に必ず -d を付けた admsetup コマンドを実行して，TPBroker の登録を解除してください。
- 既に登録されている ADM サービス名を指定した場合は，KFCB29045-E のメッセージを出力してセットアップを中止します。サービス名を変更する場合は，アンセットアップが必要です。
- ADM の複数登録機能を使用した環境で，$ADMSPOOL をカレントディレクトリとして，-r オプションを指定して「admsetup -d」を実行した場合，%ADMSPOOL% ディレクトリを削除できません。その後，-r オプションを指定して「admsetup -d」を実行しても，KFCB29079-E のメッセージが出力され，%ADMSPOOL% ディレクトリが削除できない状態になります。その場合，-r オプションを指定せずに「admsetup -d」を実行しても %ADMSPOOL% ディレクトリは削除されません。手動で %ADMSPOOL% ディレクトリを削除してください。
- 「admsetup -c 定義ファイル -r サービス名」で構築した運用支援機能の実行環境を削除する場合，必ず「admsetup -d -r サービス名」で削除してください。-r オプションを指定せずに「admsetup -d」で削除した場合，意図していない運用支援機能の実行環

境を削除する場合があります。

admstart（TPBroker の開始）

形式

admstart [-f]

機能

ADM デーモンに対してトリガをかけ，プロセス監視定義ファイルに設定された監視対象プロセスの起動および監視を開始します。

正常開始の場合は，プロセス監視定義ファイル中に設定された正常開始用のプロセスを，再開始の場合は，再開始用のプロセスを，強制正常開始の場合は，強制正常開始用のプロセスをそれぞれ起動します。

このコマンドは，tsstart コマンドで開始したトランザクションサービスでは使用できません。

オプション

-f

TPBroker が強制正常開始します。このオプションを指定しない場合は，前回の TPBroker の終了モードによって再開始または正常開始します。

表示形式

```
prompt> admstart
KFCB29031-I admstart successful.
```

注意事項

- ADM デーモンが起動していない場合は，このコマンドはメッセージを出力して終了します。
- このコマンドは，ADM デーモンに対してシステムの開始を要求します。プロセス監視定義ファイルに設定されたプロセスを起動および監視します。
- このコマンドは，ADM デーモンがコマンド受信可能状態の場合に有効です。プロセス監視定義ファイルの設定によって，デーモンプロセスがコマンド受信可能状態になるまでに時間が掛かる場合があります。
- このコマンドは，プロセス監視定義ファイルに指定されたプロセスの起動の完了を待つため，時間が掛かる場合があります。
- このコマンドで開始させた TPBroker を終了させるには，admstop コマンドを使用してください。

admstartprc（プロセスの起動と監視の開始）

形式

admstartprc -i 識別子 [識別子]... [-o コマンド名 [-d]] | [-f|-r] [-p]

機能

プロセス監視定義ファイルに設定されたプロセスを起動し，ADM デーモンに対し，起動したプロセスの監視を要求します。

また，プロセス監視定義ファイルに設定されていないプロセスを起動し，そのプロセスを監視します。

このコマンドは，admstart コマンドで TPBroker を開始させたときだけ使用できます。tsstart コマンドで開始したトランザクションサービスでは使用できません。

オプション

-i 識別子

プロセス監視定義ファイルに設定した識別子を指定します。ただし，プロセス監視定義ファイルに設定された OTS 実行環境の識別子は指定できません。プロセス監視定義ファイルに設定していないプロセスを新たに起動する場合は，ユニークな識別子（4 文字以上 32 文字以下の英数字）を指定してください。識別子は，直接起動のプロセスの場合は最初の文字が英字，間接起動のプロセスの場合は最初の文字が数字になります。

識別子は 64 個まで指定できます。この場合は，-o オプションとの併用はできません。複数の識別子を指定した場合は，指定された識別子に対するプロセスが一括して起動します。

-o コマンド名

引数で指定されたコマンドが実行され，そのプロセスが監視されます。-i オプションで指定した識別子がプロセス監視定義ファイルに設定されている場合は，-o オプションで指定したコマンドは無視され，プロセス監視定義ファイルで指定されたコマンドが起動されます。実行するコマンドがオプションを持つ場合は，「"」（引用符）で囲んで指定してください。指定できるコマンド名の長さは，オプションも含めて 255 文字以内です。

-d

-o オプションで，プロセス監視定義ファイルに設定されていないプロセスの起動要求をした場合だけに有効です。この場合，起動したプロセスが監視中に異常終了したときに，TPBroker がダウンします。

-f

プロセス監視定義ファイルに設定されたプロセスが強制正常起動します。プロセス監視

定義ファイルに強制正常開始用のプロセスの指定がない場合は，正常開始用のプロセスが起動します。

-r

プロセス監視定義ファイルに設定されたプロセスが再起動します。プロセス監視定義ファイルに再開始用のプロセスの指定がない場合は，正常開始用のプロセスが起動します。

-p

複数のプロセスを一括して起動する場合に，プロセスの起動を並列化します。

表示形式

```
prompt> admstartprc -i otsd
KFCB29071-I Process started (ID=otsd, pid=123 )
```

注意事項

- このコマンドは，実行した環境に設定された環境変数 TPSPOOL，および環境変数 ADMSPOOL と同じ設定で登録した ADM デーモンが起動していない場合は，このコマンドはメッセージを出力して終了します。
- このコマンドで動的にプロセス監視の対象に参加させたプロセスは，デーモンプロセスの子プロセスとして生成されます。
- このコマンドは，ADM デーモンがコマンド受信可能状態の場合に有効です。プロセス監視定義ファイルの設定によって，デーモンプロセスがコマンド受信可能状態になるまでに時間が掛かる場合があります。
- このコマンドは，プロセス監視定義ファイルに指定されたプロセスの起動の完了を待つため，時間が掛かる場合があります。
- このコマンドで，プロセス監視定義ファイルに指定していない間接起動方式のプロセスを起動および監視することはできません。
- 起動停止順序は admstartprc または admstopprc コマンドに指定された順とします。ただし -p オプションを付けた場合の起動停止の完了は admstartprc または admstopprc コマンドに指定された順にはなりません。プロセス間の関連づけは持たせません。

admstat（TPBroker の稼働情報表示）

形式

admstat

機能

TPBroker の現在の稼働情報を標準出力に出力します。

表示形式

```
prompt> admstat
VER-REV      : 05-12
PID          : 256
TPDIR        : C:¥TPBroker
ADMSPOOL     : C:¥TPBroker¥spool
START TIME   : 2001/05/09 15:21:33
HOSTNAME     : host01
IP-ADDR      : 111.11.11.11
PORT-NO      : 20058
CONF-MODE    : AUTO
MAX PRC NUM  : 100
```

注意事項

このコマンドは，ADM デーモンがコマンド受信可能状態の場合に有効です。プロセス監視定義ファイルの設定によって，デーモンプロセスがコマンド受信可能状態になるまでに時間が掛かる場合があります。

admstop（TPBroker の終了）

形式

admstop [-f | -fr]

機能

ADM デーモンにトリガをかけ，監視中のプロセスを定義の指定に従って停止させ，TPBroker を終了します。

このコマンドは，admstart コマンドで TPBroker を開始させたときだけ使用できます。tsstart コマンドでトランザクションサービスを開始させたときは使用できません。

オプション

-f

TPBroker が強制終了します。このオプションを指定した場合は，プロセス監視定義ファイル中の強制停止用コマンドが実行され，監視中の各プロセスが停止します。

-fr

TPBroker を強制終了します。このオプションを指定した場合は，プロセス監視定義ファイル中の強制停止用コマンドが実行され，監視中の各プロセスが停止します。ただし，-fr オプション付きの admstop コマンドが実行された時点のプロセス監視情報が保存され，次回 admstart コマンドが実行されると，この保存されたプロセス監視情報を基にプロセスが起動します。このため，コマンドで動的に追加したプロセスなど前回起動されていたプロセスすべてを，次回の再開始時に起動または監視の対象にできます。

すべてのオプションを省略した場合は，TPBroker が正常終了します。プロセス監視定義ファイル中の正常停止用コマンドが実行され，監視中の各プロセスが停止します。

表示形式

```
prompt> admstop
KFCB29031-I admstop successful.
```

注意事項

- このコマンドで停止および監視終了できるプロセスは，プロセス監視定義ファイルに記述されたプロセスか，または admstartprc コマンドで動的に監視に参加させたプロセスです。C++ 版の場合，tpadm_start_monitor 関数で動的に参加させたプロセスは，監視対象から外されますが，プロセスは停止しません。
- デーモンプロセスが起動していない場合は，このコマンドはメッセージを出力して終了します。
- このコマンドは，ADM デーモンがコマンド受信可能状態の場合に有効です。プロセス監視定義ファイルの設定によって，デーモンプロセスがコマンド受信可能状態になるまでに時間が掛かる場合があります。

- このコマンドの実行環境に設定されている環境変数 TPSPOOL，および環境変数 ADMSPOOL と同じ値で登録された ADM デーモンは停止します。
- 運用環境定義 /ADM/set_conf_mode が "AUTO" の場合，admstop コマンドの実行によって，TPBroker は一時的に停止しますが，すぐに再開始します。このため，アンインストール前などに TPBroker を終了させる場合は，運用定義 /ADM/set_conf_mode を "MANUAL" に変更したあと，admstop コマンドを実行してください。

admstopprc（監視の終了とプロセスの停止）

形式

admstopprc {-i 識別子 [識別子]...｜[-o コマンド名]}｜[-f] [-p]

機能

引数で指定された識別子に対応したプロセスを，ADM デーモンのプロセス監視の対象から外し，そのプロセスを停止します。

このコマンドは，admstart コマンドで TPBroker を開始させたときだけ使用できます。tsstart コマンドで開始した OTS 環境では使用できません。tsstart コマンドでトランザクションサービスを開始させたときは使用できません。

オプション

-i 識別子

プロセス監視定義ファイルまたは admstartprc コマンドで指定した識別子を指定します。ただし，プロセス監視定義ファイルに設定された OTS 実行環境の識別子は指定できません。

識別子は 64 個まで指定できます。この場合は，-o オプションとの併用はできません。複数の識別子を指定した場合は，指定された識別子に対するプロセスが一括して停止します。

-o コマンド名

admstartprc コマンドの -o オプションで動的に監視対象に参加させたプロセスを停止させる場合には，-o オプションでそのプロセスを停止させるためのコマンドを指定します。このオプションを指定しない場合は，識別子で指定されたプロセスがシステムコール（UNIX の場合は kill()，Windows の場合は TerminateProcess()）で直接停止します。

実行するコマンドがオプションを持つ場合は，「"」（引用符）で囲んで指定してください。指定できるコマンド名の長さは，オプションも含めて 255 文字以内です。コマンド名のパスには，スペースを含むディレクトリおよびファイル名を指定できません。

-f

識別子で指定したプロセスが強制停止します。これは，プロセス監視定義ファイルに設定されたプロセスだけに有効で，プロセス監視定義ファイル中で指定した強制停止用コマンドで停止します。プロセス監視定義ファイル中にコマンドの指定がない場合は，システムコール（UNIX の場合は kill()，Windows の場合は TerminateProcess()）でプロセスが直接停止します。

-p

複数のプロセスを一括して停止する場合に，プロセスの停止を並列化します。

表示形式

```
prompt> admstopprc -i otsd
KFCB29072-I Process stopped (ID=otsd, pid=123 )
```

注意事項

- このコマンドで停止および監視終了できるプロセスは，プロセス監視定義ファイルに記述されたプロセスか，または admstartprc コマンドで動的に監視に参加させたプロセスです。C++ 版の場合，tpadm_start_monitor 関数で動的に参加させたプロセスは，このコマンドを使用しても停止できません。
- このコマンドの実行環境に設定されている環境変数 TPSPOOL，および環境変数 ADMSPOOL と同じ値で登録された ADM デーモンが起動していない場合は，このコマンドはメッセージを出力して終了します。
- 監視しているプロセスがない場合は，このコマンドを発行してもプロセスを停止させることはできません。
- このコマンドは，ADM デーモンがコマンド受信可能状態の場合に有効です。プロセス監視定義ファイルの設定によって，ADM デーモンがコマンド受信可能状態になるまでに時間が掛かる場合があります。
- 起動停止順序は admstartprc または admstopprc コマンドに指定された順とします。ただし -p オプションを付けた場合の起動停止の完了は admstartprc または admstopprc コマンドに指定された順にはなりません。また，プロセス間の関連づけは持たせません。

trnctxsv（トランザクションコンテキストサーバの起動(Java)）

形式

trnctxsv [-Dvbroker.se.iiop_tp.scm.iiop_tp.listener.port= ポート番号] [-Dvbroker.se.iiop_tp.scm.iiop_tp.dispatcher.threadStackSize= スレッドスタックサイズ] [-Dvbroker.orb.isNTService=true]

機能

トランザクションコンテキストサーバを起動し，トランザクションコンテキストサービスを開始します。Java アプリケーションを実行する前に，トランザクションコンテキストサーバを起動しておく必要があります。

障害対策のため，TPBroker の運用支援機能から起動することをお勧めします。

このコマンドは，tsstart コマンドでトランザクションサービスを開始させたときだけ使用できます。admstart コマンドで開始した TPBroker では使用できません。

オプション

-Dvbroker.se.iiop_tp.scm.iiop_tp.listener.port= **ポート番号**

TCS が使用する TCP ポート番号を指定します。ポート番号を固定したい場合に指定してください。このオプションを指定しない場合は，OS によって自動的に割り当てられたポート番号を使用します。

-Dvbroker.se.iiop_tp.scm.iiop_tp.dispatcher.threadStackSize= **スレッドスタックサイズ**

TCS のスレッドスタックサイズを指定します。このオプションを指定しない場合は，適切なスレッドスタックサイズを使用します。このオプションは，通常は指定する必要はありません。

-Dvbroker.orb.isNTService=true

Windows のログオフ時に TCS プロセスが停止しません。このオプションを指定しない場合は，Windows のログオフ時に TCS プロセスが停止します。運用支援機能などで Windows サービスとして TCS を運用する場合は，このオプションを指定してください。

このオプションは Windows 版だけに有効です。

表示形式

```
prompt> trnctxsv
KFCB32201-I TransactionContextServer has started. pid=37106
```

注意事項

- トランザクションコンテキストサーバを起動する前に，tsstart コマンドが正常に終了

し，TPBroker が開始していることが前提です。また，トランザクションコンテキストサーバが起動していない場合，トランザクションは実行できません。
- トランザクションコンテキストサーバプロセスが停止するまで，このコマンドは制御を戻しません。
- OTS 環境ごとに一つだけトランザクションコンテキストサーバを起動できます。すでにトランザクションコンテキストサーバが起動している場合は，このコマンドはすぐに制御を戻します。

tscommit（トランザクションのコミット）

形式

tscommit {-t ｜ -T トランザクショングローバル識別子} [-f]

機能

指定したトランザクションブランチを強制的にコミットします。指定できるトランザクションブランチは，tslstrn コマンドで表示されるトランザクションの状態が PREPARED 状態であり，かつ HOLD 欄が wait のものだけです。

tscommit コマンドは，グローバルトランザクションを構成している各トランザクションブランチが何らかの要因（通信障害など）でトランザクションを決着できない場合に入力します。このとき，ほかのトランザクションとの不整合を発生させないために，グローバルトランザクション内のほかのトランザクションブランチもコミットしてください。通信障害が発生している場合，トランザクションブランチ間の連絡が完了するまでトランザクションを終了できません。このとき，-f オプションを指定するとトランザクションを強制的に終了できます。通信障害が一時的な場合，-f オプションを指定しないでコマンドを実行してください。

オプション

-t

コマンドを実行したノードのトランザクションマネージャが管理しているトランザクションで，PREPARED 状態であり，かつ HOLD 欄が wait のすべてのトランザクションのコミットを受け付けます。また，コミットするトランザクションに関する情報が標準出力に出力されます。ここで出力される情報は，tslstrn コマンドと同じ情報です。

-T **トランザクショングローバル識別子**

〜 <16 文字の英数字 >

指定されたトランザクショングローバル識別子を持つトランザクションが，PREPARED 状態であり，かつ HOLD 欄が wait であればコミットを受け付けます。また，コミットするトランザクションに関する情報が標準出力に出力されます。ここで出力される情報は，tslstrn コマンドと同じ情報です。

トランザクショングローバル識別子は，-t オプション付きの tslstrn コマンドで知ることができます。

-f

トランザクションを強制終了します。

表示形式

```
prompt> tscommit -T e06d36c602000000
GID              BID              STATUS   HOLD PID TID
e06d36c602000000 ff7236c600000001 PREPARED wait   0   0
```

tsdefremove（定義パラメタの削除）

形式

tsdefremove 定義キー名 定義パラメタ名

機能

tsdefvalue コマンドで設定されたシステム環境定義の定義パラメタを削除します。

このコマンドは，誤った名称の定義パラメタを作ってしまった場合や定義パラメタが不要になった場合などに使用します。

引数

定義キー名

定義キーの名称です。

定義パラメタ名

定義パラメタの名称です。

例

```
tsdefremove /OTS/RM/Oracle_XA xa_open_string_info
```

tsdefvalue（定義パラメタへの指定値の設定）

形式

tsdefvalue 定義キー名 定義パラメタ名 {-a｜-i｜-s} 指定値 ...

機能

指定されたシステム環境定義の定義キー下の定義パラメタに値を設定します。

指定された名称の定義パラメタが存在しない場合，新しく定義パラメタを生成して値を設定します。

引数

定義キー名

定義キーの名称です。

定義パラメタ名

定義パラメタの名称です。

-a

指定値タイプは文字列配列です。スペースを含む場合は「"」（引用符）で囲む必要があります。

-i

指定値タイプは整数です。

-s

指定値タイプは文字列です。スペースを含む場合は「"」（引用符）で囲む必要があります。

指定値

定義の指定値です。

例

```
tsdefvalue /OTS/TM max_crm_branch_count -i 32
tsdefvalue /ADM set_conf_mode -s "AUTO"
tsdefvalue /OTS completion_process_env -a "ENV1=value1"
"ENV2=value2"
```

注意事項

次の条件に該当する定義パラメタを指定した場合，メッセージ KFCB31033-W を標準エラー出力とメッセージログファイルに出力します。

- TPBroker で規定されていない定義パラメタ名
- 定義キー名との組み合わせが不一致の定義パラメタ名
- 指定値タイプとの組み合わせが不一致の定義パラメタ名

tsedapt（API トレースファイル解析（C++））

形式

tsedapt API トレースファイル名

機能

API トレースファイルを解析し，結果を出力します。解析結果はスレッド単位に出力します。

このコマンドは，Cosminexus TPBroker ではサポートしていません。

引数

API トレースファイル名

解析する API トレースファイルの名称を指定します。

表示形式

```
    ******************************
(1)  Thread id = 174
    ******************************
(2) [OUT  ] CosTransactions::Current::begin()
(3)         Serial number = 0
(4)         EntryTime = 1999/09/17 13:39:18.123
(5)         ReturnTime = 1999/09/17 13:39:20.234
    [OUT  ] CosTransactions::Current::get_status()
(6)         Return Status = CosTransactions::Status::StatusActive
            Serial number = 1
            EntryTime = 1999/09/17 13:39:22.543
            ReturnTime = 1999/09/17 13:39:23.012
    [EXCEP] CosTransactions::Current::commit()
(7)         in boolean report_heuristics = 00000001
(8)         Exception name =IDL:omg.org/CORBA/NO_MEMORY:1.0
            Serial number = 2
            EntryTime = 1999/09/17 13:39:25.091
            ReturnTime = 1999/09/17 13:39:30.784
    ******************************
      Thread id = 218
    ******************************
                 :
                 :
```

（凡例）

(1)　Thread id：スレッド ID

(2)　[IN　]：API 入り口トレース

[OUT]：API 出口トレース

[EXCEP]：API 内での例外発生トレース

CosTransactions::…：メソッド名

(3)　Serial number：スレッド内通番

(4)　EntryTime：API 入り口トレース取得時刻

(5)　ReturnTime：API 出口トレース取得時刻
(6)　Return Status：API の戻り値
(7)　in…：API の引数
(8)　Exception name：API 内で発生した例外名

(2) から (5) までの範囲がトレース単位となります。また，API によって出力内容が異なります。

なお，解析する API トレースファイルが指定されていない場合は，エラーメッセージ「Usage: tsedapt filename」が出力され，用法が表示されます。

tsedtrntrc（トランザクショントレースの出力（UNIX））

形式

tsedtrntrc -f トレースファイル名

機能

TPBroker のトランザクショントレースを標準出力に出力します。

なお，これは UNIX 版だけに提供されるコマンドです。

オプション

-f トレースファイル名

指定されたファイルのトレースを標準出力に出力します。

表示形式

```
DATE        TIME         PID     GID              BID              RES
2001/12/06 14:52:05.021 33222   0123456789abcdef 0123456789abcdef B
1998/02/27 22:03:02.553 12345   0123456789abcdef 0123456789abcdef C
1998/02/27 22:04:03.112 23451   23456789abcdef01 0000000001000000 RB
1998/02/27 22:20:06.132 34512   56789abcdef01234 0000000001000000 HH
1998/02/27 22:34:01.123 2345    789abcdef0123456 0000000001000000 HM
```

（凡例）

DATE：トランザクションブランチが開始または決着した日付

TIME：トランザクションブランチが開始または決着した時刻

PID：トランザクションブランチを扱っているプロセスのプロセス ID

GID：トランザクションブランチのグローバル識別子

BID：トランザクションブランチの識別子

RES：トランザクションブランチの決着処理の結果
- B：開始しました。
- C：コミットしました。
- RB：ロールバックしました。
- HH：決着処理の結果がわかりません（HeuristicHazard）。
- HM：コミットおよびロールバックしました（HeuristicMixed）。
- HC：ヒューリスティックにコミットしました（HeuristicCommit）。
- HR：ヒューリスティックにロールバックしました（HeuristicRollback）。
- RO：トランザクションブランチはリードオンリーです。
- --：すでに決着処理が終了しています。
- **：どのように決着処理されたかわかりません。

tskeycreate（定義キーの生成）

形式

tskeycreate 定義キー名

機能

指定された名称のシステム環境定義の定義キーを生成します。

引数

定義キー名

定義キーの名称です。

例

```
tskeycreate /OTS/RM/Oracle_XA
```

tskeyremove（定義キーの削除）

形式

tskeyremove 定義キー名

機能

tskeycreate コマンドで生成されたシステム環境定義の定義キーを削除します。

このコマンドは，誤った名称の定義キーを作ってしまった場合や定義キーが不要になった場合などに使用します。

ただし，誤って必要なシステムの定義キーを指定しても削除されてしまうので注意してください。

引数

定義キー名

定義キーの名称です。

例

```
tskeyremove /OTS/RM/Oracle_XA
```

tslnkrm（リソースマネージャの登録・削除（C++））

形式

tslnkrm -d 削除するリソースマネージャ名 [, 削除するリソースマネージャ名]...[-l][-m][-f]

tslnkrm -n [-l][-m][-f]

tslnkrm -a 追加するリソースマネージャ名 [, 追加するリソースマネージャ名]...
　-s リソースマネージャスイッチ名 [, リソースマネージャスイッチ名]...
　-o ' リソースマネージャ関連オブジェクト名 [リソースマネージャ関連オブジェクト名]...' [, ' リソースマネージャ関連オブジェクト名 [リソースマネージャ関連オブジェクト名]...']
　[-e] [-r] [-i] [-l] [-m] [-f]

機能

OTS で使用するリソースマネージャを追加または削除します。また，回復デーモン（rcvd），決着デーモン（complete），標準 XA-switch list ファイル（tp_allrm.o）を再作成します（Windows 版の場合，標準 XA-switch list ファイルは tp_allrm.obj という名前になります）。

tslnkrm コマンドは OTS が動作している間は実行できません。また，再開始待ちの場合は，tslnkrm コマンドに -f オプションを指定してください。ただし，-f オプションを指定した場合には，トランザクションサービスを再開始することはできません。次回の開始モードは正常開始モードになります。

tslnkrm コマンドは C++ コンパイラを使用します。tslnkrm コマンドがコンパイラを実行できるように，環境変数 PATH にコンパイラがあるディレクトリをあらかじめ追加しておいてください（Windows 版の場合，環境変数 INCLUDE にヘッダファイルがあるディレクトリを，環境変数 LIB にライブラリがあるディレクトリをあらかじめ追加しておいてください）。

このコマンドは，Cosminexus TPBroker ではサポートしていません。

オプション

-d 削除するリソースマネージャ名

～ <1 ～ 32 文字の英数字 >

削除するリソースマネージャの名称を指定します。

このオプションで指定したリソースマネージャに対しては，リソースマネージャスイッチ名，リソースマネージャ関連オブジェクト名を指定する必要はありません。複数のリソースマネージャ名を指定する場合は，リソースマネージャ名とリソースマネージャ名

との間を「,」（コンマ）で区切ってください。

-l

tslnkrm コマンドの実行経過の詳細が標準出力に出力されます。

-m

このオプションを指定してもしなくても，OTS デーモンはマルチスレッドで動作します。これによってトランザクションの決着処理や回復処理が同時実行できるようになるため，トランザクション処理全体の性能を向上できます。このオプションは互換性のために残してあります。

OTS で使用するすべてのリソースマネージャの XA インタフェースがマルチスレッドに対応している必要があります。TPBroker は，マルチスレッドに対応していない XA インタフェースを持ったリソースマネージャを使用できません。

-f

OTS の状態に関係なく，強制的に実行されます。

ただし，デーモンを再作成するため，OTS が動作している間は実行されません。このオプションは，トランザクションサービスを正常終了以外（強制終了，異常終了）で終了したあと，使用するリソースマネージャを変更してトランザクションサービスを正常開始する場合にだけ指定してください。このオプションを指定した場合は，次回の開始モードは正常開始になります。

-n

登録しているリソースマネージャは変更されないで，デーモンの再作成だけが実行されます。

-a **追加するリソースマネージャ名**

〜 <1 〜 32 文字の英数字 >

追加するリソースマネージャの名称を指定します。

このオプションで指定したリソースマネージャに対しては，リソースマネージャスイッチ名，リソースマネージャ関連オブジェクト名を指定する必要があります。複数のリソースマネージャ名を指定する場合は，リソースマネージャ名とリソースマネージャ名との間を「,」（コンマ）で区切ってください。

-s **リソースマネージャスイッチ名**

〜 < 先頭が英字または _（アンダースコア）で始まる 1 〜 32 文字の英数字 >

追加するリソースマネージャのスイッチ名を指定します。

スイッチ名は，追加するリソースマネージャのマニュアルを参照してください。複数のリソースマネージャスイッチ名を指定する場合は，リソースマネージャスイッチ名とリ

ソースマネージャスイッチ名との間を「,」（コンマ）で区切ってください。

リソースマネージャスイッチ名とリソースマネージャ名は指定した順に対応します。

-o リソースマネージャ関連オブジェクト名

〜＜英数字＞

追加するリソースマネージャに関連のあるオブジェクトファイル（XA インタフェース用のオブジェクトファイル）の名称を指定します。

リソースマネージャ関連オブジェクト名は，追加するリソースマネージャのマニュアルを参照してください。

一つのリソースマネージャに対して複数のリソースマネージャ関連オブジェクトを指定できます。複数のリソースマネージャ関連オブジェクト名を指定する場合は，リソースマネージャ関連オブジェクト名とリソースマネージャ関連オブジェクト名との間をスペースで区切ってください。複数のリソースマネージャに対するリソースマネージャ関連オブジェクト名を指定する場合は，一つのリソースマネージャに対するリソースマネージャ関連オブジェクト名の集まりを「'」（アポストロフィ）で囲み，それぞれの集まりの間を「,」（コンマ）で区切ってください。

リソースマネージャ関連オブジェクト名とリソースマネージャ名は指定した順に対応します。

リソースマネージャ関連オブジェクトはデーモンの再作成のときに使われるため，それらを絶対パス名で指定してください。

-e

-s オプションで指定されたリソースマネージャスイッチ名が XA-Switch 構造体へのポインタのシンボルとして参照されます。

このオプションは，Windows 版だけに有効です。

-r

追加するリソースマネージャが動的登録をサポートしている場合に，このオプションを指定します。これによって，リソースマネージャから ax_reg および ax_unreg 関数を使用できます。

-i

-o オプションで指定されたオブジェクトをトランザクション制御用共用ライブラリにリンクします。

-i オプション付きで登録されたオブジェクトは，tslnkrm コマンドで生成される標準トランザクション制御用共用ライブラリと，tsmkobj コマンドで生成されるトランザクション制御用共用ライブラリにリンクされます。

なお，このオプションは Windows 版では自動的に付加されます。

例

```
tslnkrm -a Oracle_XA -s xaosw -o
'-L/export/Oracle/oracle816/app/oracle/product/8.1.6/lib -lclntsh'
tslnkrm -d Oracle_XA
```

tslogcat（メッセージログの出力）

形式

tslogcat [メッセージログファイル名]

機能

TPBroker またはトランザクションコンテキストサーバのメッセージを標準出力に出力します。

引数

メッセージログファイル名

標準出力に出力させたいログファイルを指定します。この引数を省略した場合は，$TPSPOOL/log/otslog ファイルのメッセージが出力されます。

トランザクションコンテキストサーバのメッセージを出力させるには，$TPSPOOL/log/trnctxsvlog ファイルを指定してください。

TPBroker で出力されたログファイル以外のファイルを指定すると，コマンドは何も出力しないで終了します。

表示形式

```
prompt> tslogcat
2001/05/23 14:38:27 24207 1 KFCB31444-I Starting OTS Daemon.
pid=24207  date=Wed May 23 14:38:27 2001

2001/05/23 14:38:31 24208 1 KFCB31462-I Starting Completion Daemon.
pid=24208  date=Wed May 23 14:38:31 2001

2001/05/23 14:38:31 24208 1 KFCB31463-I Completion Daemon has
started.

2001/05/23 14:38:31 24209 1 KFCB31465-I Starting Recovery Daemon.
pid=24209  date=Wed May 23 14:38:31 2001

2001/05/23 14:38:32 24209 1 KFCB31466-I Recovery Daemon has started.

2001/05/23 14:38:33 24207 1 KFCB31449-I Transaction Service has
started.
```

tslsconf（定義パラメタの表示）

形式

tslsconf [-c] [定義キー名]

機能

現在の TPBroker のシステム環境定義についての情報を表示します。環境変数 TPSPOOL および引数の定義キー名で指定されたキーを基に定義パラメタを表示します。これらの情報は標準出力に出力されます。引数の定義キー名が指定されていない場合は，すべての定義パラメタが出力されます。

オプション

-c

定義パラメタをチェックします。引数の定義キー名が指定されている場合，指定された定義キー名下の定義パラメタだけをチェックします。次の条件に該当する定義パラメタを指定した場合，メッセージ KFCB31033-W を標準エラー出力とメッセージログファイルに出力します。

- TPBroker で規定されていない定義パラメタ名
- 定義キー名との組み合わせが不一致の定義パラメタ名
- 指定値タイプとの組み合わせが不一致の定義パラメタ名

このオプションを省略した場合は，現在の TPBroker のすべてのシステム環境定義についての情報を表示します。

引数

定義キー名

出力したい定義キーの名称を指定します。指定しない場合は，すべてのシステム環境の情報を出力します。

表示形式

```
prompt> tslsconf /OTS/TM
/OTS/TM
/OTS/TM/process_count                    32
/OTS/TM/max_crm_branch_count             8
/OTS/TM/set_status_group_num             1
/OTS/TM/set_status_write_mode            "none"
/OTS/TM/set_recovery_mode                0
```

tslsfs（TPBroker ファイルシステムの内容表示（UNIX））

形式

tslsfs [-H] [-L] [-t | -u] TPBroker ファイルシステム領域名 [/TPBroker ファイル名]

tslsfs [-x] TPBroker ファイルシステム領域名 [/TPBroker ファイル名]

機能

TPBroker ファイルシステムの内容を標準出力に出力します。

コマンド引数に TPBroker ファイルシステム領域名だけを指定した場合は，指定した TPBroker ファイルシステム内にあるすべての TPBroker ファイルの内容をアルファベット順で出力します。TPBroker ファイル名も指定した場合は，指定した TPBroker ファイルの内容を出力します。

なお，これは UNIX 版だけに提供されるコマンドです。

オプション

-H

表示する情報にヘッダが付けられ，ファイル名のアルファベット順に，縦方向に表示されます。

-L

ファイルのロック状態が，ファイル名のアルファベット順に，縦方向に表示されます。

-t

最終更新日時が最近のものから順に，TPBroker ファイルシステムの内容が表示されます。

-H オプションまたは -L オプションと，-t オプションを同時に指定した場合，表示内容の順序は，-t オプションの指定が有効になります。

-u

最終アクセス日時が最近のものから順に，TPBroker ファイルシステムの内容が表示されます。

-H オプションまたは -L オプションと，-u オプションを同時に指定した場合，表示内容の順序は，-u オプションの指定が有効になります。

-x

ファイル名だけがアルファベット順に，横方向に表示されます。

以上のオプションを省略した場合は，TPBroker ファイルシステムの内容がファイル名の

アルファベット順に，縦方向に表示されます。

引数

TPBroker ファイルシステム領域名

～<パス名>

TPBroker ファイルシステムがあるキャラクタ型スペシャルファイル名を指定します。

TPBroker ファイル名

TPBroker ファイル名を指定します。

表示形式

● -H オプションを指定した場合

```
MODE   UID     GID      RSIZE  RNUM  TIME    FILE
aabbcc dd...dd ee...ee  ffff   ggggg hh...hh ii...ii
```

● -H オプションと -L オプションを指定した場合

```
MODE   UID     GID      PID   L  TIME    FILE
aabbcc dd...dd ee...ee  pppp  q  hh...hh ii...ii
```

（凡例）

aa：所有者に対するアクセス権です。
- r：読み込み権があります。
- w：書き込み権があります。
- -：読み込み権，および書き込み権がありません。

bb：グループに対するアクセス権です。
- r：読み込み権があります。
- w：書き込み権があります。
- -：読み込み権，および書き込み権がありません。

cc：他者に対するアクセス権です。
- r：読み込み権があります。
- w：書き込み権があります。
- -：読み込み権，および書き込み権がありません。

dd...dd：所有者名（9 文字以内）です。

ee...ee：所有者のグループ名（9 文字以内）です。

ffff：レコード長です。

ggggg：レコード数です。

hh...hh：最終更新日時です。「時：分△月△日△年」の形式で出力されます。

ii...ii：TPBroker ファイル名です。

pppp：ロックを掛けているプロセスのプロセス ID です。ロックが掛けられてい

ない場合は，- が表示されます

q：ロック状態の識別フラグです。

E：占有ロックが掛けられています。

S：共有ロックが掛けられています。

-：ロックが掛けられていません。

tslsrm（リソースマネージャ情報の表示（C++））

形式

tslsrm [-o ファイル名 [, ファイル名] ...] [-s]

機能

トランザクションサービス，アプリケーションプログラム，またはトランザクション制御用オブジェクトに登録されているリソースマネージャの情報を標準出力に出力します。

このコマンドは，Cosminexus TPBroker ではサポートしていません。

オプション

-o ファイル名

～＜パス名＞

指定したファイルに登録されているリソースマネージャの情報が標準出力に出力されます。

指定できるファイルは，アプリケーションプログラムのロードモジュール，またはトランザクション制御用オブジェクトファイルだけです。複数のファイル名を指定する場合は，ファイル名とファイル名との間を「,」（コンマ）で区切ってください。

-s

トランザクションサービスに登録されているリソースマネージャの情報が標準出力に出力されます。

すべてのオプションを省略した場合は，-s オプションが指定されたものとみなされます。

表示形式

```
prompt> tslsrm
File Name      RM-name      Attributes XA-switch  Objs
System         OTSCRM       e          *tpotsxasw
               DmyRM1       i          dmyrmsw1   C:¥dmyrm.obj
```

tslnkrm コマンドに -e オプションを指定して登録したリソースマネージャでは，リソースマネージャスイッチ名の先頭に「*」（アスタリスク）が付きます。

Attributes 欄の記号の意味を次に示します。

e：tslnkrm コマンドに -e にオプションを指定して登録したリソースマネージャ（Windows）

i：tslnkrm コマンドに -i オプションを指定して登録したリソースマネージャ

tslstrn（トランザクションの状態表示）

形式

tslstrn {-t ｜ -T トランザクショングローバル識別子 } [-c]

機能

OTS が管理しているトランザクションに関する情報，またはトランザクションブランチ数を表示します。

オプション

-t

すべてのトランザクションに関する情報が表示されます。

-T **トランザクショングローバル識別子**

指定したグローバル識別子を持つトランザクションに関する情報が表示されます。

-c

同時に指定したオプションに従って，トランザクションブランチ数が表示されます。

表示形式

```
prompt> tslstrn -t
GID              BID              STATUS             HOLD   PID   TID
e06d36c618000000 0000000074000000 COMMIT_ONE_PHASE  other  398   397
e06d36c618000000 e06d36c675000000 PREPARE           other  285   332
```

（凡例）

GID：トランザクションブランチのグローバル識別子

BID：トランザクションブランチの識別子

STATUS：トランザクションブランチの処理状態

ACTIVE：トランザクション処理中
BEGINNING：トランザクション開始
COMMIT：コミット処理中
COMMIT_ONE_PHASE：1 相コミット処理中
FORGETTING：トランザクション完了済み
HEURISTIC_COMMIT：コミットでヒューリスティック発生
HEURISTIC_FORGETTING：ヒューリスティック状態への決着指示
HEURISTIC_ROLLBACK：ロールバックでヒューリスティック発生
IDLE：待ち
PREPARE：プリペア処理中
PREPARED：サーバ側決着指示待ち

ROLLBACK：ロールバック処理中
ROLLBACK_ONLY：ロールバック待ち
SUSPENDED：トランザクション中断中

HOLD：トランザクションブランチの処理状態
other：その他
recover：回復デーモンでの回復待ち
wait：サーバ側回復処理で，replay_completion が 1 回以上失敗している状態。
決着コマンド受け付けは可能な状態。

PID：トランザクションブランチを扱っているプロセスのプロセス ID

TID：トランザクションブランチを扱っているスレッドのスレッド ID

tsmkfs（TPBroker ファイルシステムの初期設定（UNIX））

形式

tsmkfs -s セクタ長 -n 容量 -l 最大ファイル数

[-v TPBroker ファイルシステム名] スペシャルファイル名

機能

指定したハードディスクのパーティションを，TPBroker ファイルシステム用に初期設定します。初期設定は，TPBroker ファイルシステムとしてパーティションを割り当てるときに一度だけ行います。

tsmkfs コマンドを実行できるのは，スーパーユーザ，またはキャラクタ型スペシャルファイルの所有者です。

なお，これは UNIX 版だけに提供されるコマンドです。

オプション

-s セクタ長

TPBroker ファイルシステムを構築するハードディスクのセクタ長を指定します。

-n 容量

〜 ((1 〜 2047))(単位：メガバイト)

TPBroker ファイルシステムとして割り当てる容量をメガバイトで指定します。

-l 最大ファイル数

〜 ((1 〜 4096))

TPBroker ファイルシステム内に作成するファイル数の上限を指定します。

キャラクタ型スペシャルファイルを環境変数 TPFS に設定する場合は 2 以上，環境変数 ADMFS に設定する場合は 3 以上，環境変数 TPFS および ADMFS の両方に設定する場合は 5 以上を指定します。

-v TPBroker ファイルシステム名

〜 <1 〜 8 文字の TPBroker ファイル名 >

TPBroker ファイルシステムに付ける名称を指定します。

このオプションの指定を省略した場合は，TPBroker ファイルシステムに名称が付けられません。

引数

スペシャルファイル名

～<パス名>

初期化するスペシャルファイルの名称を指定します。

指定するファイルはキャラクタ型スペシャルファイルです。

例

```
tsmkfs -s 1024 -n 10 -l 3 -v TPB01 /raw/tpb_raw
```

注意事項

初期設定時，容量としてディスクボリューム，またはパーティションの容量よりも大きな値を指定すると，そのパーティションに物理的に続くパーティションを破壊することがあります。このため，TPBroker 管理者は，TPBroker ファイルシステムの容量を正確に計算してディスクを割り当ててください。また，UNIX ファイルシステムとして使用しているパーティションを指定しないでください。

tsmkobj（トランザクション制御用オブジェクトファイルの作成（C++））

形式

tsmkobj -o トランザクション制御用オブジェクト名 [-r リソースマネージャ名 [, リソースマネージャ名]...] [-l]

機能

アプリケーションプログラムで使用するリソースマネージャの XA-switch を含んだトランザクション制御用オブジェクトファイルとトランザクション制御用共用ライブラリを作成します。

オブジェクトファイルは，$TPSPOOL/XA 下に「トランザクション制御用オブジェクト名 .o」という名称で作成されます（Windows 版の場合，「トランザクション制御用オブジェクト名 .obj」という名称で作成されます）。このオブジェクトファイルと各リソースマネージャが提供するリソースマネージャ関連オブジェクトファイルをアプリケーションプログラムにリンクしてください。リンクすると，トランザクションサービス下でリソースマネージャにアクセスするトランザクションを実行できます。

アプリケーションプログラムに tslnkrm コマンドが作成した「tp_allrm.o」（Windows 版の場合は「tp_allrm.obj」）をリンクする場合は，tsmkobj コマンドで新たにオブジェクトファイルを作成する必要はありません。CRM 以外のリソースマネージャを使用しない場合は，$TPSPOOL/XA にある「tp_crm.o」をアプリケーションプログラムにリンクしてください（Windows 版の場合，「tp_crm.obj」をリンクしてください）。アプリケーションプログラムからトランザクションサービスの機能を使用するには，このファイルをリンクする必要があります。

tsmkobj コマンドは C++ コンパイラを使用します。tsmkobj コマンドがコンパイラを実行できるように，環境変数 PATH にコンパイラがあるディレクトリをあらかじめ追加しておいてください（Windows 版の場合，環境変数 INCLUDE にヘッダファイルがあるディレクトリを，環境変数 LIB にライブラリがあるディレクトリをあらかじめ追加しておいてください）。

このコマンドは，Cosminexus TPBroker ではサポートしていません。

オプション

-o トランザクション制御用オブジェクト名

～ <1 ～ 12 文字の英数字 >

トランザクション制御用オブジェクトファイルの名称を指定します。

拡張子を指定する必要はありません。

-r リソースマネージャ名

～ <1 ～ 32 文字の英数字 >

アプリケーションプログラムからアクセスするリソースマネージャの名称を指定します。

トランザクションサービスに登録されていないリソースマネージャは指定できません。どのリソースマネージャが登録されているかは，tslsrm コマンドを使用して確かめることができます。

複数のリソースマネージャ名を指定する場合は，リソースマネージャ名とリソースマネージャ名の間を「,」（コンマ）で区切ってください。

このオプションを省略した場合は，トランザクションサービスに登録されているすべてのリソースマネージャを指定したものとみなされます。

-l

tsmkobj コマンドの実行経過の詳細が標準出力に出力されます。

例

```
tsmkobj -o myOracle -r Oracle_XA
```

tsrasget（障害調査資料の採取（UNIX））

形式

tsrasget [-b][-m][-s] -d 出力ディレクトリ [付加情報 [付加情報]...]

機能

TPBroker の障害調査資料を採取します。このコマンドの実行ログを，引数で指定したディレクトリ下に出力します。デフォルトで ORB 障害調査資料，ADM 障害調査資料，OTS 障害調査資料を取得します。

このコマンドの実行時には，TPBroker で必要な環境変数をあらかじめ設定しておいてください。また，このコマンドは root 権限を持つユーザで実行してください。

なお，これは UNIX 版だけに提供されるコマンドです。

オプション

-b

ORB 障害調査資料を取得しない場合に指定します。

-m

ADM 障害調査資料を取得しない場合に指定します。

-s

OTS 障害調査資料を取得しない場合に指定します。

-d 出力ディレクトリ

障害調査資料を出力するディレクトリを指定します。このディレクトリ下に，ORB 障害調査資料，ADM 障害調査資料，および OTS 障害調査資料が出力されます。引数に指定する出力ディレクトリは絶対パスで記述してください。

引数

付加情報

このコマンドで取得する障害調査資料以外で，取得したい資料を指定します。付加情報は，ORB 障害調査資料として採取されます。付加情報としてファイルを指定した場合は，そのファイルが取得されます。ディレクトリを指定した場合は，そのディレクトリ以下が取得されます。

付加情報は複数指定できます。複数指定する場合は，スペースで区切ってください。

例

```
tsrasget -b -m -d /ots/home/tpbroker
```

注意事項

- 取得情報容量は，$TPSPOOL/*，$ADMSPOOL/*，$TPFS/*，$ADMFS/*，$TPDIR/conf/*，$VBROKER_ADM/../log/*，および $VBROKER_ADM/../logj/* を加えた値以上必要になります。
- このコマンドは root 権限を持つユーザで実行してください。

tsrollback（トランザクションのロールバック）

形式

tsrollback {-t ｜ -T トランザクショングローバル識別子 [-F] } [-f]

機能

指定したトランザクションブランチを強制的にロールバックします。指定できるトランザクションブランチは，tslstrn コマンドで表示されるトランザクションの状態がPREPARED 状態であり，かつ HOLD 欄が wait のものだけです。

tsrollback コマンドは，グローバルトランザクションを構成している各トランザクションブランチが何らかの要因（通信障害など）でトランザクションを決着できない場合に入力します。このとき，ほかのトランザクションとの不整合を発生させないために，グローバルトランザクション内のほかのトランザクションブランチもロールバックしてください。通信障害が発生している場合，トランザクションブランチ間の連絡が完了するまでトランザクションを終了できません。このとき，-f オプションを指定すると，トランザクションを強制的に終了できます。通信障害が一時的な場合，-f オプションは指定しないでコマンドを実行してください。

オプション

-t

コマンドを実行したノードのトランザクションマネージャが管理しているトランザクションで，PREPARED 状態であり，かつ HOLD 欄が wait のすべてのトランザクションのロールバックを受け付けます。また，ロールバックするトランザクションに関する情報が標準出力に出力されます。ここで出力される情報は，tslstrn コマンドで出力されるものと同じです。

-T **トランザクショングローバル識別子**

〜 <16 文字の英数字 >

指定されたトランザクショングローバル識別子を持つトランザクションが，PREPARED 状態であり，かつ HOLD 欄が wait であればロールバックを受け付けます。また，ロールバックするトランザクションに関する情報が標準出力に出力されます。ここで出力される情報は，tslstrn コマンドと同じ情報です。

トランザクショングローバル識別子は，-t オプション付きの tslstrn コマンドで知ることができます。

-F

-T オプションで指定したトランザクションにスペリア（呼び出し元のトランザクションブランチ）がない場合，そのトランザクションがロールバックされます。

-F オプションを指定した場合，-T オプションに指定されたトランザクショングローバル識別子を持つトランザクションが，PREPARED 状態であり，かつ HOLD 欄が other であれば処理を受け付けます。このトランザクションのうち，スペリアがすでになければロールバックが行われます。スペリアがまだある場合は，コマンドによるロールバックは行われません。スペリアから決着指示があったときに，その指示に従って決着処理が行われます。

-F オプションと -f オプションを同時に指定した場合は，-f オプションは無視されます。

-f

トランザクションを強制終了します。

表示形式

```
prompt> tsrollback -T e06d36c603000000
GID              BID              STATUS   HOLD PID TID
e06d36c603000000 ff7236c600000003 PREPARED wait   0   0
```

tssetup（TPBroker のセットアップ）

形式

tssetup {-d | [-u] [-i] [-n]}

機能

OTS 環境を初期化します。また，作業用のディレクトリを作成，初期化，および TPBroker の定義パラメタを初期化します。

オプション

-d

OTS 環境が削除されます。環境変数 TPSPOOL および TPFS で設定したディレクトリは削除されます。

-u

リソースマネージャ定義 /OTS/RM/…/xa_open_string_info および /OTS/RM/…/xa_close_string_info の定義値を表示または出力する条件を，次のように変更します。このオプションは UNIX 版だけに有効です。

- tslsconf，tsstat コマンド実行時の定義値の表示
  root 権限を持つユーザまたは tssetup コマンド実行ユーザが実行した場合，定義値を表示します。
  それ以外のユーザが実行した場合，定義値を表示しません。
- XA トレースファイルへの定義値の出力
  定義値を出力しません。

-i

環境変数 TPFS に設定されたディレクトリ，またはキャラクタ型スペシャルファイルを複数の OTS 環境で共有する場合に指定します。このオプションを指定した場合と指定しない場合の動作を次の表に示します。

表 9-4　-i オプションを指定する場合と指定しない場合の動作（tssetup コマンドの場合）

動作	-i オプションあり	-i オプションなし
$TPFS ディレクトリがある場合の tssetup コマンドの動作	正常終了	異常終了
$TPFS ディレクトリの作成タイミング	tsstart コマンド実行時	tssetup コマンド実行時
-d オプションを指定して tssetup コマンドを実行した場合に $TPFS ディレクトリを削除するかどうか	削除しない	削除する

-i オプションを指定すると $TPFS ディレクトリは削除されないため，どの環境からも参照されなくなった $TPFS ディレクトリは OS のコマンドなどで削除してください。また，-d オプションを指定して tssetup コマンドを実行すると，$TPSPOOL ディレクトリは削除されるため，-d オプションを指定して tssetup コマンドを実行したあとに，-i オプションを指定して tssetup コマンドを実行する場合は，環境変数 TPFS に環境変数 TPSPOOL とは別のパスを明示的に設定してください。

-n

tssetup コマンド実行後の最初の OTS の起動を正常開始にします。$TPFS ディレクトリを共用ディスクや TPBroker ファイルシステムに配置すると，トランザクションステータスファイルの情報が前回の OTS 停止時の状態で残る場合があります。このオプションは，OTS 環境を作成し直す場合でトランザクションステータスファイルの情報を引き継ぐ必要がないときに指定してください。

表示形式

```
prompt> tssetup
All Rights Reserved, Copyright (C) 1996,2004, Hitachi,Ltd.

KFCB31023-I Starting system setup configurations.

KFCB31002-I tssetup successful.
```

注意事項

-d オプションを指定した場合，TPBroker の作業用ディレクトリが削除されます。TPBroker の動作中は，-d オプションを付けた tssetup コマンドは実行しないでください。

- Windows 版 05-19 以降は tssetup コマンドが tstpbsetup コマンドへ名称が変更されました。互換性のため tssetup コマンドがを使用したい場合は，%TPDIR%¥bin¥objs¥tssetup.exe を %TPDIR%¥bin¥tssetup.exe にコピーして使用してください。

tsstart（トランザクションサービスの開始）

形式

tsstart [-r|-f]

機能

トランザクションサービスを開始します。

tsstart コマンドは ORB が利用できるときに有効となります。したがって，tsstart コマンドは ORB が利用できないことを検知すると，メッセージを出力して終了します。

tsstart コマンドで OTS が起動すると，OTS はトランザクションステータスファイルをチェックし，未決着トランザクションを回復し，すべてのトランザクションサービスを開始します。

このコマンドは，admstart コマンドで開始した TPBroker では使用できません。

オプション

-r

トランザクションをチェックし，未決着トランザクションを回復します。この場合，OTS は tscommit，tsrollback，tsstop コマンドなどを受け付けますが，アプリケーションプログラムは新しいトランザクションを生成しません。

-f

前回の終了形態に関係なく，OTS が強制的に正常開始します。この場合，OTS は初期化された状態から開始するため，前回仕掛かり中のトランザクションの決着処理は行われません。

すべてのオプションを省略した場合は，OTS を開始します。前回の終了形態から開始種別が決定されます。前回正常終了していない場合には，前回未決着のトランザクションの決着が行われます。

表示形式

```
prompt> tsstart
KFCB31486-I tsstart successful.
```

注意事項

- tsstart コマンド実行時に，前回の OTS のセションに参加したプロセスが残っている場合は，そのプロセスを強制停止します。
- このコマンドで開始させたトランザクションサービスを終了させるには，tsstop コマンドを使用してください。

tsstat（OTS の状態表示）

形式

tsstat [-c | -r リソースマネージャ名 [, リソースマネージャ名]...]

機能

現在の OTS 稼働状態の情報を表示します。tsstat コマンドは次に示すオプション以外のオプションが指定されても無視します。

オプション

-c

すべてのリソースマネージャ情報，一部のトランザクションマネージャ情報，環境変数 TPDIR，TPFS，TPSPOOL およびホスト名を表示します。OTS がオフラインの場合，このオプションは無視されます。

-r リソースマネージャ名

リソースマネージャ名に指定されたリソースマネージャ情報を表示します。複数のリソースマネージャを指定することもできます。OTS がオフラインの場合，または -c オプションが指定されている場合，このオプションは無視されます。

すべてのオプションを省略した場合は，OTS がオンラインかどうかを表示します。OTS がオンラインの場合，KFCB31496-I が表示され，OTS がオフラインの場合，KFCB31497-I が表示されます。

戻り値

0

OTS がオンラインの場合に返されます。

0 以外の値

警告，エラーが発生した場合，または OTS がオフラインの場合に返されます。次の原因が考えられます。

- 環境変数 TPDIR が設定されていません。
- スプールディレクトリが存在しません。
- システムコールレベルでエラーが発生しました。

表示形式

-c オプションを指定した場合，次の三つの情報を表示します。

リソースマネージャ情報

OTS 実行環境に登録されているすべてのリソースマネージャ情報です。

```
/OTS/RM/OTSCRM
/OTS/RM/OTSCRM/set_xa_open_scope                "process"
/OTS/RM/OTSCRM/set_xa_rmfail_action             "retry1"
/OTS/RM/OTSCRM/set_xa_rmerr_action              "retry1"
/OTS/RM/OTSCRM/DMN/xa_open_string_info          ""
/OTS/RM/OTSCRM/DMN/xa_close_string_info         ""
```

トランザクションマネージャ情報

OTS 実行環境に登録されている一部のトランザクションマネージャ情報です。

```
/OTS/TM/process_count                             32
/OTS/TM/max_crm_branch_count                      8
/OTS/TM/set_status_write_mode                     "none"
/OTS/TM/set_recovery_mode                         0
```

その他

OTS 実行環境に登録されている環境変数およびホスト名の情報です。

```
TPDIR:                          /opt/TPBrokerV5
TPSPOOL:                        /opt/TPBrokerV5/otsspool
TPFS:                           /opt/TPBrokerV5/tpfs
HOSTNAME:                       prius
```

tsstatfs（TPBroker ファイルシステムの状態表示（UNIX））

形式

tsstatfs [-w] TPBroker ファイルシステム領域名

機能

指定した TPBroker ファイルシステムの状態を標準出力に出力します。

なお，これは UNIX 版だけに提供されるコマンドです。

オプション

-w

TPBroker ファイル管理領域の開始位置を表示します。このオプションは AIX 版だけに有効です。

引数

TPBroker ファイルシステム領域名

～＜パス名＞

TPBroker ファイルシステムがあるキャラクタ型スペシャルファイル名を指定します。

表示形式

```
Sector length [B]                        : aa...aa
Total size of user area [KB]             : bb...bb
Free size of user area [KB]              : cc...cc
Maximum file size [KB]                   : dd...dd
Maximum number of files                  : ee...ee
Number of created files                  : ff...ff
Number of files which can be created     : gg...gg
Number of free areas                     : hh...hh
Initialize user of file system           : ii...ii
Initialize time of file system           : jj...jj
File system name                         : kk...kk
Start point of TPBroker file control area: ll...ll※
```

注※　-w オプションを指定した場合に表示されます。

（凡例）

aa...aa：

tsmkfs コマンドの -s オプションで指定したセクタ長です。単位はバイトです。

bb...bb：

TPBroker ファイルシステム中でユーザに割り当てられた領域の総容量です。単位はキロバイトです。

cc...cc：

ユーザに割り当てられた領域の中で，未使用（TPBroker ファイルとして割り当てられていない）のものの容量です。単位はキロバイトです。

dd...dd：

現在，一つのファイルとして確保できる容量の最大値です。単位はキロバイトです。

ee...ee：

tsmkfs コマンドの -l オプションで指定した，作成できるファイルの上限数です。

ff...ff：

すでに作成されたファイルの数です。

gg...gg：

作成できるファイル数（最大作成可能ファイル数－作成済みファイル数）です。

hh...hh：

連続していない空き領域の総数です。

ii...ii：

TPBroker ファイルシステムを初期設定したユーザのログイン名です。

jj...jj：

TPBroker ファイルシステムを初期設定した時刻です。「曜日△月△日△時：分：秒△年（西暦）」の形式で出力されます。

kk...kk：

tsmkfs コマンドで指定した TPBroker ファイルシステム名です。

ll...ll：

TPBroker ファイル管理領域の開始位置です。

tsstop（トランザクションサービスの終了）

形式

tsstop [-f [1|2]]

機能

新しいグローバルトランザクションの生成を禁止し，トランザクションサービスを終了します。

このコマンドは，tsstart コマンドでトランザクションサービスを開始させたときだけ使用できます。admstart コマンドで TPBroker を開始させたときは使用できません。

オプション

-f 1

新しいグローバルトランザクションとサブオーディネートブランチの生成が禁止されます。OTS は動いているすべてのトランザクションの決着を待ちます。

-f 2

すべてのトランザクションサービスがすぐに終了します。この場合，動いているすべてのトランザクションは OTS によって処理がサスペンドされ，OTS はトランザクションの決着を待ちません。

なお，「-f」だけが指定された場合は，-f 2 オプションが指定されたものとみなされます。

すべてのオプションを省略した場合は，トランザクションサービスを正常終了します。また，新しいグローバルトランザクションの生成が禁止されます。OTS は動いているすべてのトランザクションの決着を待ちます。

表示形式

```
prompt> tsstop
KFCB31487-I tsstop successful.
```

tsstoptrnctxsv（トランザクションコンテキストサーバの終了（Java））

形式

tsstoptrnctxsv [-f]

機能

トランザクションコンテキストサーバを終了します。オプションの指定がない場合，新しいトランザクションの生成を抑止し，実行中のすべてのトランザクションが決着するまでトランザクションコンテキストサーバの終了を待ちます。

このコマンドは，tsstart コマンドでトランザクションサービスを開始させたときだけ使用できます。admstart コマンドで TPBroker を開始させたときは使用できません。

オプション

-f

実行中のすべてのトランザクションの決着を待たないでロールバックさせ，トランザクションコンテキストサーバを強制的に終了します。タイムアウトしない，またはタイムアウト値が 5 分以上であるトランザクションを終了する場合にこのオプションを指定してください。

例

```
tsstoptrnctxsv -f
```

注意事項

トランザクションコンテキストサーバプロセスが停止するまで，このコマンドは制御を戻しません。ただし，5 分以内に停止しない場合，トランザクションコンテキストサーバを終了しないで制御を戻します。

tstrnsts（トランザクション稼働統計情報の出力）

形式

tstrnsts

機能

トランザクション稼働統計情報を標準出力に出力します。

表示形式

```
Start Time                                            2001/12/20 20:15:31
Transaction Time Out                                  3

Toproot information
        Total Transaction Branch                      100015
        Live Transaction Branch                       10
        Transaction Commit                            99998
        Transaction Rollback                          1
        Transaction Heuristic                         1
        Transaction Read Only                         5

Subroot information
        Total Transaction Branch                      22
        Live Transaction Branch                       10
        Transaction Commit                            10
        Transaction Rollback                          1
        Transaction Heuristic                         1
        Transaction Read Only                         0

Subordinate information
        Total Transaction Branch                      25
        Live Transaction Branch                       10
        Transaction Commit                            11
        Transaction Rollback                          1
        Transaction Heuristic                         1
        Transaction Read Only                         2
```

（凡例）

Start Time：TPBroker の OTS 機能が開始された時刻

Transaction Time Out：タイムアウトしたトランザクションブランチの数

Toproot information：トップトランザクションブランチのルートブランチ情報

Subroot information：サブトランザクションブランチのルートブランチ情報

Subordinate information：上記二つ以外のトランザクションブランチ情報

Total Transaction Branch：トランザクションブランチの累計

Live Transaction Branch：稼働中のトランザクションブランチの数

Transaction Commit：コミットしたトランザクションブランチの数

Transaction Rollback：ロールバックしたトランザクションブランチの数

Transaction Heuristic：ヒューリスティックに決着したトランザクションブランチの数

Transaction Read Only：リードオンリーのトランザクションブランチの数

注意事項

- このコマンドはノード内の TPBroker にだけ適用されます。
- このコマンドは TPBroker の OTS 機能が起動している間だけ使用できます。
- Transaction Time Out に表示される数は，TPBroker がタイムアウトを検知したときにカウントされます。タイムアウトしたトランザクションブランチも決着したトランザクションブランチ数に含まれます。
- トップトランザクションブランチで rollback_only() を発行した場合でも，トップトランザクションブランチがリソースマネージャにアクセスしていなければ，リードオンリーのトランザクションブランチにカウントされることがあります。
- OTS Fast Path Option を使用している場合，サーバコール時にサーバプロセスが異常終了しても，トップトランザクションブランチがリソースマネージャにアクセスしていなければ，リードオンリーのトランザクションブランチにカウントされることがあります。
- システムダウンが発生した場合，ロールバックしたトランザクションブランチに関しては出力されないことがあります。
- TPBroker を再開始した場合，トランザクションの稼働統計情報は初期状態になります。
- 次に示す値は 4294967295 を超えると 0 に戻ります。
    - タイムアウトしたトランザクションブランチの数
    - トランザクションブランチの累計
    - 稼働中のトランザクションブランチの数
    - コミットしたトランザクションブランチの数
    - ロールバックしたトランザクションブランチの数
    - ヒューリスティックに決着したトランザクションブランチの数
    - リードオンリーのトランザクションブランチの数

10 障害対策

この章では，TPBroker の障害を分類し，それぞれの障害が発生した場合の対処方法を説明します。また，障害情報の取得方法についても説明します。

10.1　アプリケーションプログラムの障害

アプリケーションプログラムの障害が発生したときの対処を説明します。

アプリケーションプログラムの詳細については，マニュアル「TPBroker　プログラマーズガイド」を参照してください。

10.1.1　異常終了するとき

TPBroker の処理が異常終了したのか，またはアプリケーションプログラムの処理が異常終了したのかをデバッガなどで調査してください。

TPBroker が異常終了した場合，先にメッセージが出力されているときは「11.2　メッセージ一覧」に記載されている対策内容に従って対処してください。

アプリケーションプログラムの処理が異常終了した場合は，アプリケーションプログラムを修正したあと，アプリケーションプログラムを再開始してください。この場合，トレースファイルと，コアファイルがあればこれらを退避しておいてください。

原因が不明の場合は，「10.3　障害の解決に必要な情報」に従い，障害情報を取得し，保守員に連絡してください。

10.2　TPBroker の障害

TPBroker の障害が発生したときの対処を説明します。

10.2.1　TPBroker が正しくインストール，およびセットアップされていないとき

TPBroker のインストール，およびセットアップをやり直してください。

10.2.2　システム環境定義が誤っているとき

システム環境定義で設定する項目に不良がある場合，その不良内容がメッセージに出力されます。

システム環境定義の定義項目を見直し，正しく設定し直したあと，TPBroker を再開始してください。

10.2.3　OS の構成が TPBroker の実行環境として不適当なとき

共用メモリサイズや最大プロセス数などの OS の構成を見直し，必要なパラメタを変更したあと，TPBroker を再開始してください。

10.2.4　異常終了するとき

先にメッセージが出力されている場合は「11.2　メッセージ一覧」に記載されている対策内容に従って対処してください。

「11.2　メッセージ一覧」に記載されていないメッセージが出力され原因が不明の場合は，「10.3　障害の解決に必要な情報」に従い，障害情報を取得し，保守員に連絡してください。

UNIX 版で TPBroker の運用支援機能を使用している場合，/etc/inittab から ADM デーモンのプロセスが起動されます。このデーモンプロセスが 10 分間に 10 回連続で異常終了した場合には，TPBroker の運用支援機能は使用できなくなります。

10.2.5　TPBroker の運用コマンドが正常終了しないとき

オプション，またはコマンド引数の指定が誤っていることを示すメッセージが出力される場合は，正しく指定し直してください。「11.2　メッセージ一覧」に記載されていないメッセージが出力され原因が不明の場合は，「10.3　障害の解決に必要な情報」に従い，障害情報を取得し，保守員に連絡してください。

運用コマンドを実行できる環境が設定されていないことを示すメッセージが出力される場合は，正しい環境を設定してください。特に環境変数の設定に注意してください。

該当する運用コマンドを実行する権限がないときは，権限を持ったユーザで実行してください。

コマンドが正常終了または異常終了しないで止まった場合は，コマンドのプロセスおよび TPBroker のデーモンプロセスを強制的に停止させてください。

10.2.6　Java 実行環境で障害が発生したとき（Java）

TPBroker は，Java アプリケーションでの OTS API のログを取得します。ログを取得すると，障害調査に利用できます。

ログを取得するには，TPBroker がインストールされている必要があります。TPBroker のインストールの詳細については，「2.　TPBroker の環境設定」を参照してください。

Java アプリケーションでの OTS API のログを取得するには，TpCosOTS インタフェースの set_property メソッドで log_trace_level を設定して，トレース情報を取得できるようにする必要があります。詳細は，マニュアル「TPBroker　プログラマーズガイド」を参照してください。ログは次の方法で表示できます。

- 環境変数 TPSPOOL が設定されている場合，次のコマンドを発行します。
  ```
  tslogcat $TPSPOOL/log/trnctxsvlog
  ```
- 環境変数 TPSPOOL が設定されていない場合，次のコマンドを発行します。
  ```
  tslogcat $TPDIR/otsspool/log/trnctxsvlog
  ```

10.2.7　Cosminexus の J2EE トランザクションで障害が発生したとき（Cosminexus TPBroker）

Cosminexus TPBroker の場合，メッセージの出力先によって対処方法が異なります。

次に示す出力先にメッセージが出力された場合，メッセージへの対処方法については，「11.2　メッセージ一覧」を参照してください。「11.2　メッセージ一覧」に記載されていないメッセージが出力されて原因が不明の場合は，「10.3　障害の解決に必要な情報」を参照し，障害情報を取得して保守員に連絡してください。

- syslog（UNIX）
- イベントログ（Windows）
- OTS メッセージログファイル
- トランザクションコンテキストサーバのメッセージログファイル

次に示す出力先にメッセージが出力された場合，詳細なログ出力個所およびメッセージへの対処方法については，マニュアル「Cosminexus アプリケーションサーバ V8 システ

ム構築・運用ガイド」および「Cosminexus アプリケーションサーバ V8 メッセージ 3 KECX/KEOS20000-29999/KEUC-KFRM 編」を参照してください。マニュアル「Cosminexus アプリケーションサーバ V8 メッセージ 3 KECX/KEOS20000-29999/KEUC-KFRM 編」に記載されていないメッセージが出力されて原因が不明の場合は，「10.3　障害の解決に必要な情報」を参照し，障害情報を取得して保守員に連絡してください。

- Cosminexus Component Container のメッセージログ（J2EE サーバの稼働ログ）
- Cosminexus Component Container のユーザログ（J2EE サーバのユーザ出力ログ）

10.3　障害の解決に必要な情報

障害が発生した場合，障害情報として，$TPSPOOL，$TPFS，$ADMSPOOL，$VBROKER_ADM/…/log，$VBROKER_ADM/…/logj および $ADMFS ディレクトリ下の全ファイルを退避してください。UNIX 版の場合，tsrasget コマンドで障害情報を取得できます。$TPSPOOL ディレクトリには OTS 機能から出力されたファイル，$ADMSPOOL ディレクトリには運用支援機能から出力されたファイルがあります。プロセス異常終了によってコアファイルまたはダンプファイルが出力される場合は，コアファイルまたはダンプファイル，およびコアファイルまたはダンプファイルに対応した実行形式ファイルも退避してください。運用支援機能の監視プロセスのコアファイルの取得先，及びファイル名は「6.2.1　プロセス監視の概要」をご参照ください。

障害の解決に必要な情報を次の二つの表に示します。これらの情報は，障害発生後，システムを再開始すると上書きされるものもありますので注意が必要です。

10.3.1　UNIX 版の場合

表 10-1　障害の解決に必要な情報（UNIX 版の場合）

取得先	取得情報	注意事項
コンソールメッセージ	TPBroker が出力するシステム情報	OS 起動時には出力されません。
コアファイル （$ADMSPOOL/core） （$TPSPOOL/errinfo/save）	TPBroker の関連プロセスのデータ，スタック情報	残しておきたいコアファイルは，必要に応じてコピーしてください。またコアファイルに対応した実行形式ファイルも残すようにしてください。
トレースファイル （$TPSPOOL/trace/traceXXX） （$ADMSPOOL/traceXXX） （XXX：プロセス番号 + 生成時刻）	TPBroker のモジュールトレース情報	残しておきたいトレースファイルは必要に応じてコピーしてください。
トランザクションコンテキストサーバのメッセージログファイル（Java） （$TPSPOOL/log/trnctxsvlog）	トランザクションコンテキストサーバのメッセージログ	—
トランザクションステータスファイル （$TPFS/tmsts）	トランザクションステータス情報	TPBroker ファイルシステムを使用している場合は，OS が提供する dd コマンドを使用して情報を退避してください。
TPBroker システムステータスファイル （$TPFS/status.tmd） （$ADMFS/admsts） （$ADMFS/admmon） （$ADMFS/admproc）	TPBroker システムステータス情報	TPBroker ファイルシステムを使用している場合は，OS が提供する dd コマンドを使用して情報を退避してください。

取得先	取得情報	注意事項
TPBroker 保守ファイル （$TPSPOOL/tmdproc） （$TPSPOOL/tmshm） （$TPSPOOL/.command） （$ADMSPOOL/.admcom） （$ADMSPOOL/admshmsys）	TPBroker 保守情報	－
TPBroker メッセージログファイル （$ADMSPOOL/log/admlog）	TPBroker の運用支援機能が出力したメッセージ	－
syslog ファイル	TPBroker が出力したメッセージ	－
標準出力および標準エラー出力 （$ADMSPOOL/log/stdlog1，stdlog2）	TPBroker および監視対象プロセスが標準出力および標準エラー出力に出力したメッセージ	－
ORB トレースファイル （$VBROKER_ADM/../log） （$VBROKER_ADM/../log/mdltrc） （$VBROKER_ADM/../log/comtrc） （$VBROKER_ADM/../log/hgttrc）	エラーログ，モジュールトレース，通信トレース，OSAgent のバーボースログ	$VBROKER_ADM/../log はデフォルトの出力先です。設定によって出力先を変更できます。 障害発生時にはコピーして退避してください。
Java 実行環境の ORB トレースファイル（Java） （$VBROKER_ADM/../logj） （$VBROKER_ADM/../logj/mdltrc） （$VBROKER_ADM/../logj/comtrc） （Cosminexus TPBroker） （$"Cosminexus Component Container の作業ディレクトリ "/ejb/" サーバ名 "/logs/TPB/logj） （$"Cosminexus Component Container の作業ディレクトリ "/ejb/" サーバ名 "/logs/TPB/logj/mdltrc） （$"Cosminexus Component Container の作業ディレクトリ "/ejb/" サーバ名 "/logs/TPB/logj/comtrc）	エラーログ，モジュールトレース，通信トレース，メッセージログ（TPBroker 05-15 以降）	$VBROKER_ADM/../logj はデフォルトの出力先です。設定によって出力先を変更できます。 障害発生時にはコピーして退避してください。 （Cosminexus TPBroker） "Cosminexus Component Container の作業ディレクトリ "/ejb/" サーバ名 "/logs/TPB/logj はデフォルトの出力先です。 Cosminexus Component Container の設定によって出力先を変更できます。 障害発生時にはコピーして退避してください。
（Cosminexus TPBroker） Cosminexus Component Container のメッセージログ （" 作業ディレクトリ "/ejb/" サーバ名 "/logs/cjmessage?.log）	TPBroker メッセージ	作業ディレクトリ，サーバ名については，マニュアル「Cosminexus アプリケーションサーバ V8 システム構築・運用ガイド」および「Cosminexus アプリケーションサーバ V8 メッセージ 3 KECX/KEOS20000-29999/KEUC-KFRM 編」を参照してください。

取得先	取得情報	注意事項
（Cosminexus TPBroker） Cosminexus Component Container の ユーザログ （" 作業ディレクトリ "/ejb/" サーバ名 "/ logs/user_out?.log）	TPBroker メッセージ	作業ディレクトリ，サーバ名については，マニュアル「Cosminexus アプリケーションサーバ V8 システム構築・運用ガイド」および「Cosminexus アプリケーションサーバ V8 メッセージ 3 KECX/ KEOS20000-29999/ KEUC-KFRM 編」を参照してください。
（Cosminexus TPBroker） Cosminexus Component Container の EJB コンテナの保守情報 （" 作業ディレクトリ "/ejb/" サーバ名 "/ logs/cjejbcontainer?.log）	TPBroker トレース情報	作業ディレクトリ，サーバ名については，マニュアル「Cosminexus アプリケーションサーバ V8 システム構築・運用ガイド」および「Cosminexus アプリケーションサーバ V8 メッセージ 3 KECX/ KEOS20000-29999/ KEUC-KFRM 編」を参照してください。
（Cosminexus TPBroker） リソースマネージャのログ	リソースマネージャのログ	ご使用のリソースマネージャのマニュアルを参照して取得してください。
監視対象プロセスコアファイル （$ADMSPOOL/errinfo/save）	TPBroker の監視下にあるプロセスのデータ，スタック情報	－
API トレースファイル （$TPSPOOL/aptrace）	OTS アプリケーションが出力する API トレース	Cosminexus TPBroker では不要です。
XA トレース （$TPSPOOL/log/XAtrace, XAtrace.bak）	XA 関数発行時のエラー情報	－
例外情報ファイル （$TPSPOOL/log/iinfo0，iinfo1, iinfo2）	OTS アプリケーション使用時の例外情報	－
OTS メッセージログファイル （$TPSPOOL/log/otslog）	OTS のデーモンプロセス，コマンドおよび OTS アプリケーションが出力したメッセージ	－
運用支援機能の監視対象プロセスのカレントディレクトリ （$ADMSPOOL/tmp/home/" 識別子 "） （識別子：プロセス監視定義ファイルに設定した識別子，または，admstartprc コマンドで指定した識別子）	監視対象プロセスがカレントディレクトリへ出力したファイル	残しておきたいファイルは必要に応じてコピーしてください。
運用支援機能の保守情報 （$ADMSPOOL_logN）（N=1，2，3）	運用支援機能の保守情報	－

取得先	取得情報	注意事項
プロセス監視定義ファイル	使用しているプロセス監視定義ファイル	−
トランザクショントレースファイル（$TPSPOOL/trcinfo/trnspool/dcopltrc）	OTS アプリケーションおよび OTS のデーモンプロセスが出力するトランザクション情報	−
環境変数一覧	env コマンドの出力結果	−
システム環境定義一覧	tslsconf コマンドの出力結果	−
接続リソースマネージャ情報一覧	tslsrm コマンドの出力結果	−

（凡例）　−：該当する内容がないことを表します。

10.3.2　Windows 版の場合

表 10-2　障害の解決に必要な情報（Windows 版の場合）

取得先	取得情報	注意事項
コンソールメッセージ	TPBroker が出力するシステム情報	OS 起動時には出力されません。
ダンプファイル	Dr.Watson が出力するクラッシュダンプファイル	アプリケーションエラーが 2 回以上発生すると，このファイルは上書きされます。必要に応じてコピーしてください。また，クラッシュダンプファイルに対応した実行形式ファイルも残すようにしてください。 Dr.Watson の仕様，設定については，Dr.Watson for Windows Help を参照してください。
トレースファイル （%TPSPOOL%¥trace¥traceXXX） （%ADMSPOOL%¥traceXXX） （XXX：プロセス番号 + 生成時刻）	TPBroker のモジュールトレース情報	残しておきたいトレースファイルは必要に応じてコピーしてください。
トランザクションステータスファイル（%TPFS%¥tmsts）	トランザクションステータス情報	−
TPBroker システムステータスファイル （%TPFS%¥status.tmd） （%ADMFS%¥admsts） （%ADMFS%¥admmon） （%ADMFS%¥admproc）	TPBroker システムステータス情報	−

取得先	取得情報	注意事項
TPBroker 保守ファイル （%TPSPOOL%¥tmdproc） （%TPSPOOL%¥tmdshm） （%TPSPOOL%¥tmshm） （%TPSPOOL%¥.command） （%ADMSPOOL%¥.admcom） （%ADMSPOOL%¥admshmsys）	TPBroker 保守情報	－
TPBroker メッセージログファイル （%ADMSPOOL%¥log¥admlog）	TPBroker の運用支援機能が出力したメッセージ	－
admlaunch のメッセージログファイル（Java） （%ADMSPOOL%¥log¥admlaunchlog）	admlaunch コマンドが出力したメッセージ	－
イベントログ	TPBroker が出力したメッセージ	－
標準出力および標準エラー出力 （%ADMSPOOL%¥log¥stdlog1, stdlog2）	TPBroker および監視対象プロセスが，標準出力および標準エラー出力に出力したメッセージ	－
ORB トレースファイル （%VBROKER_ADM%¥..¥log） （%VBROKER_ADM%¥..¥log¥mdltrc） （%VBROKER_ADM%¥..¥log¥comtrc）	エラーログ，モジュールトレース，通信トレース	%VBROKER_ADM%¥..¥log はデフォルトの出力先です。設定によって出力先を変更できます。障害発生時にはコピーして退避してください。
Java 実行環境の ORB トレースファイル（Java） （%VBROKER_ADM%¥..¥logj） （%VBROKER_ADM%¥..¥logj¥mdltrc） （%VBROKER_ADM%¥..¥logj¥comtrc） （Cosminexus TPBroker） （$"Cosminexus Component Container の作業ディレクトリ "¥ejb¥" サーバ名 "¥logs¥TPB¥logj） （$"Cosminexus Component Container の作業ディレクトリ "¥ejb¥" サーバ名 "¥logs¥TPB¥logj¥mdltrc） （$"Cosminexus Component Container の作業ディレクトリ "¥ejb¥" サーバ名 "¥logs¥TPB¥logj¥comtrc）	エラーログ，モジュールトレース，通信トレース，UAP トレース（Cosminexus TPBroker には不要）	%VBROKER_ADM%¥..¥logj はデフォルトの出力先です。設定によって出力先を変更できます。 障害発生時にはコピーして退避してください。 （Cosminexus TPBroker） "Cosminexus Component Container の作業ディレクトリ "¥ejb¥" サーバ名 "¥logs¥TPB¥logj はデフォルトの出力先です。 Cosminexus Component Container の設定によって出力先を変更できます。 障害発生時にはコピーして退避してください。

取得先	取得情報	注意事項
(Cosminexus TPBroker) Cosminexus Component Container のメッセージログ (" 作業ディレクトリ "¥ejb¥" サーバ名 "¥logs¥cjmessage?.log)	TPBroker メッセージ	作業ディレクトリ，サーバ名については，マニュアル「Cosminexus アプリケーションサーバ V8 システム構築・運用ガイド」および「Cosminexus アプリケーションサーバ V8 メッセージ 3 KECX/KEOS20000-29999/KEUC-KFRM 編」を参照してください。
(Cosminexus TPBroker) Cosminexus Component Container のユーザログ (" 作業ディレクトリ "¥ejb¥" サーバ名 "¥logs¥user_out?.log)	TPBroker メッセージ	作業ディレクトリ，サーバ名については，マニュアル「Cosminexus アプリケーションサーバ V8 システム構築・運用ガイド」および「Cosminexus アプリケーションサーバ V8 メッセージ 3 KECX/KEOS20000-29999/KEUC-KFRM 編」を参照してください。
(Cosminexus TPBroker) Cosminexus Component Container の EJB コンテナの保守情報 (" 作業ディレクトリ "¥ejb¥" サーバ名 "¥logs¥cjejbcontainer?.log)	TPBroker トレース情報	作業ディレクトリ，サーバ名については，マニュアル「Cosminexus アプリケーションサーバ V8 システム構築・運用ガイド」および「Cosminexus アプリケーションサーバ V8 メッセージ 3 KECX/KEOS20000-29999/KEUC-KFRM 編」を参照してください。
(Cosminexus TPBroker) リソースマネージャのログ	リソースマネージャのログ	ご使用のリソースマネージャのマニュアルを参照して取得してください。
API トレースファイル (%TPSPOOL%¥aptrace)	OTS アプリケーションが出力する API トレース	Cosminexus TPBroker では不要です。
XA トレース (%TPSPOOL%¥log¥XAtrace, XAtrace.bak)	XA 関数発行時のエラー情報	－
例外情報ファイル (%TPSPOOL%¥log¥iinfo0，iinfo1, iinfo2)	OTS アプリケーション使用時の例外情報	－
OTS メッセージログファイル (%TPSPOOL%¥log¥otslog)	OTS のデーモンプロセス，コマンドおよび OTS アプリケーションが出力したメッセージ	－

取得先	取得情報	注意事項
トランザクションコンテキストサーバのメッセージログファイル（Java）（%TPSPOOL%¥log¥trnctxsvlog）	トランザクションコンテキストサーバのメッセージログ	－
運用支援機能の保守情報（%ADMSPOOL%_logN）（N=1，2，3）	運用支援機能の保守情報	－
接続リソースマネージャ情報一覧	tslsrm コマンドの出力結果	－
プロセス監視定義ファイル	使用しているプロセス監視定義ファイル	－

（凡例）　－：該当する内容がないことを表します。

11 メッセージ

この章では，TPBroker が出力するメッセージの形式および詳細を説明します。

11.1 メッセージの形式

メッセージが実際に出力される形式，およびこの章での説明の形式を次に示します。

11.1.1 メッセージの出力形式

メッセージの出力形式を次に示します。

```
KFCBnnnnn-X  YY.....YY.

    KFCBnnnnn：メッセージID（半角英数字9文字）
    X：エラーレベル
      E：エラー
      I：情報
      W：警告
    YY.....YY：メッセージテキスト
```

11.1.2 メッセージの記述形式

この章のメッセージの記述形式を次に示します。

メッセージ ID

メッセージテキスト

説明

[システムの処理]

システムがメッセージを出力したあとにする主な処理を示します。

[対策]

メッセージ確認時の TPBroker 管理者の処置を示します。

11.1.3 メッセージの出力先

メッセージの出力先は，標準出力，標準エラー出力，syslog（UNIX），およびイベントログ（Windows）です。

11.2　メッセージ一覧

TPBroker が出力するメッセージの一覧を示します。

メッセージの出力先は，標準出力，標準エラー出力，syslog（UNIX），およびイベントログ（Windows）です。なお，KFCB50000 ～ KFCB59999 は，UAP ログ出力機能を利用してユーザアプリケーションプログラムが出力するメッセージです。

KFCB29000-I

System has already invoked.

[システムの処理]

ADM デーモンがすでに起動しているので，何もしません。コマンドは正常終了します。

[対策]

なし。

KFCB29001-I

System has not invoked yet.

[システムの処理]

ADM デーモンが起動していないので，何もしません。コマンドは正常終了します。

[対策]

なし。

KFCB29002-E

Failure to allocate memory.

[システムの処理]

コマンドは異常終了します。

[対策]

ほかのアプリケーションプログラムを終了させてください。

KFCB29003-I

Failure to allocate memory in ADMD.

[システムの処理]

ADM デーモンが異常終了します。コマンドは異常終了します。

[対策]

ほかのアプリケーションプログラムを終了させてください。

KFCB29004-E

Failure to initialize command.

[システムの処理]

コマンドは異常終了します。

[対策]

保守員に連絡してください。

KFCB29005-E

Communication error occurred.

[システムの処理]

コマンドは異常終了します。

[対策]

1. 運用定義 /ADM/port_id_info に設定しているポート番号が，アクセスしたい ADM デーモンと同じかどうかを tslsconf コマンドで確認してください。異なっていれば運用コマンドで運用定義を変更してください。また，マシンにホスト名が設定されているかどうかを確認し，設定されていなければホスト名を設定してください。
2. ほかのアプリケーションプログラムが運用定義 /ADM/port_id_info に設定されているポート番号を使用していないかを確認してください。
3. ADM デーモンが稼働しているかどうかを確認してください。

KFCB29006-E

Failure to open system definition (aa...aa).

aa...aa：定義キー

[システムの処理]

コマンドは異常終了します。

[対策]

定義キーが設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合は tsdefvalue コマンドを入力して，システム環境定義を設定してください。

KFCB29007-E

Failure to get system definition (aa...aa).

aa...aa：定義パラメタ

［システムの処理］

コマンドは異常終了します。

［対策］

システム環境定義 aa...aa の取得に失敗しました。定義パラメタ aa...aa が正しく設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合は，tsdefvalue コマンドを入力して，システム環境定義を設定してください。

KFCB29008-E

Invalid definition (aa...aa=bb...bb) is found.

aa...aa：定義パラメタ

bb...bb：定義値

［システムの処理］

コマンドは異常終了します。

［対策］

システム環境定義 aa...aa の値が不正です。定義パラメタ aa...aa に正しい値が設定されているかどうかを tslsconf コマンドで確認してください。正しい値が設定されていない場合は，tsdefvalue コマンドを入力して，正しいシステム環境定義を設定してください。

KFCB29011-E

Function (aa...aa) returns error code (bb...bb).

aa...aa：異常を検知した関数名

bb...bb：関数のエラーコード

［システムの処理］

コマンドは異常終了します。

［対策］

ADM デーモン側で異常を検知した場合がありますので，ADM デーモンが出力したメッセージを admlogcat コマンドで確認し，そのメッセージに対する処置を行ってください。

KFCB29012-I

aa...aa

aa...aa：コマンドの使用方法

［システムの処理］

コマンドの使用方法を示して，コマンドは正常終了します。

[対策]

メッセージに従って，正しい使用方法で再度コマンドを入力してください。

KFCB29013-E

Cannot access to ADMD.

[システムの処理]

コマンドは異常終了します。

[対策]

ほかの運用コマンドが ADM デーモンにアクセスしているか，または ADM デーモンが起動していません。ほかの運用コマンドの終了を待って，再度コマンドを入力するか，または ADM デーモンが起動していない場合は起動してください（Windows 版の場合，サービス「TPBroker」を開始しておく必要があります）。運用ミスによって ADM デーモンを手動で起動すると，手動で起動された ADM デーモンは運用定義 /ADM/port_id_info を書き換え，KFCB29179-I を syslog，イベントログ，メッセージログファイルに出力します。このあと手動で起動した ADM デーモンを停止すると，OS に登録されている ADM デーモンと運用コマンドの間でポート番号の不一致が発生し，このメッセージが出力されます。この場合，運用定義 /ADM/port_id_info の値を，KFCB29179-I で出力されたシステム環境定義で指定されたポート番号に再設定してください。

KFCB29014-I

The command received invalid message format from ADMD.

[システムの処理]

コマンドは異常終了します。

[対策]

保守員に連絡してください。

KFCB29015-E

'$ADMDIR' is not set.

[システム処理]

プロセスは異常終了します。

[対策]

環境変数 TPDIR を設定して，プロセスを再起動してください。

KFCB29016-I

The specified process (aa...aa) has been already monitored.

aa...aa：指定された識別子

[システムの処理]

コマンドは正常終了します。

[対策]

なし。

KFCB29017-I

The specified process (aa...aa) has not been monitored yet.

aa...aa：指定された識別子

[システムの処理]

コマンドは正常終了します。

[対策]

なし。

KFCB29018-E

$ADMSPOOL or $ADMFS directory does not exist.

[システムの処理]

コマンドは異常終了します。

[対策]

admsetup コマンドを入力し，実行環境を構築してください。

KFCB29019-E

File (aa...aa) is not found.

aa...aa：ファイル名

[システムの処理]

コマンドは異常終了します。

[対策]

プロセス監視定義ファイルに設定されたファイル名やコマンドのオプションで指定したファイル名が正しいかを確認してください。これらのファイル名はすべて絶対パス名で設定してください。

KFCB29020-E

Number of monitored processes exceeded the value defined in the system definition. (aa...aa)

aa...aa：指定された識別子

[システムの処理]

コマンドは異常終了します。

[対策]

tsdefvalue コマンドで運用定義 /ADM/max_process_num の値を変更し，再度 ADM デーモンを起動してください。

KFCB29021-E

Configuration file does not exist.

[システムの処理]

コマンドは異常終了します。ADM デーモンは処理を続けます。

[対策]

admstart コマンドでこのメッセージが出力された場合，ADM デーモンはプロセス監視定義ファイルに何も定義されていないものとみなし，処理を開始します。TPBroker を終了させ，admsetup コマンドを再度実行して，正しいプロセス監視定義ファイルを指定してください。admreload コマンドでこのメッセージが出力された場合，プロセス監視定義ファイルを admsetup コマンド入力時に指定したパスに作成し，再度 admreload コマンドを実行してください。

KFCB29022-E

Process (aa...aa) is not defined in the configuration file.

aa...aa：指定された識別子

[システムの処理]

コマンドは異常終了します。

[対策]

プロセス監視定義ファイルに指定された識別子と，admstartprc コマンドの -i オプションに指定した識別子が一致しているかどうか確認を指定して，再度コマンドを入力してください。

KFCB29023-E

Cannot stop process (ID=aa...aa).

aa...aa：停止に失敗したプロセスの識別子

［システムの処理］

コマンドは異常終了します。

［対策］

指定された識別子に対するプロセスは，API で監視対象に参加させたプロセスなので，コマンドで停止させることはできません。

KFCB29024-E

Cannot start process (ID=aa...aa).

aa...aa：起動に失敗したプロセスの識別子

［システムの処理］

コマンドは異常終了します。

［対策］

プロセス監視定義ファイルまたはコマンドのオプションに指定されたファイルが実行形式ファイルかどうかをチェックしてください。また，16 ビットアプリケーションの場合は，起動できても監視対象に参加させることができないので，手動でプロセスを停止させてください。

KFCB29025-E

Length of specified file name is too long.

［システムの処理］

コマンドは異常終了します。

［対策］

admstartprc または admstopprc コマンドのオプションに指定されたファイル名が長過ぎます。オプションも含めて 255 文字以内のファイル名を指定してください。

KFCB29026-E

The specified key (aa...aa) does not exist.

aa...aa：指定された定義キー

［システムの処理］

コマンドは異常終了します。

［対策］

システム環境定義を tslsconf コマンドで参照し，正しい定義キー名を指定してください。

KFCB29028-E

The specified definition (aa...aa) does not exist.

aa...aa：指定された定義パラメタ

[システムの処理]

コマンドは異常終了します。

[対策]

システム環境定義を tslsconf コマンドで参照し，正しい定義パラメタを指定してください。

KFCB29029-W

Number of the invoked processes indirectly exceeded the limit(100). (aa...aa)

aa...aa：指定された識別子

[システムの処理]

コマンドは異常終了します。

[対策]

admstartprc コマンドで指定したプロセス監視定義ファイルに設定されている，間接起動方式のプロセスの起動プロセス数を見直してください。間接起動方式では 100 プロセスまで監視できます。

KFCB29030-W

Cannot monitor processes(es) by indirect activation. ID=aa...aa

aa...aa：プロセスに対する識別子

[システムの処理]

コマンドは異常終了します。

[対策]

間接起動方式で起動された監視対象プロセスが起動しているか，またはプロセス ID 取得用コマンドが正しく動作するかどうかを確認してください。監視対象プロセスが起動している場合は，ADM デーモンの監視対象外になっていますので手動でプロセスを停止させてください。監視対象プロセスが正しく起動するか，またはプロセス ID 取得用コマンドが正常に動作することを確認して，再度 admstartprc コマンドを実行してください。

KFCB29031-I

aa...aa successful.

aa...aa：コマンド名

[システムの処理]

コマンドは正常終了しました。

[対策]

なし。

KFCB29032-E

Must be super user.

[システムの処理]

コマンドは異常終了します。

[対策]

スーパーユーザで再度コマンドを実行してください。

KFCB29033-E

This system is not supported by TPBroker.

[システムの処理]

コマンドは異常終了します。

[対策]

OS 名が HP-UX の場合は "HP-UX"，Solaris の場合は "SunOS"，AIX の場合は "AIX" と出力されるようにコンピュータシステムの情報を uname コマンドで設定してください。

KFCB29034-E

TPBroker is already registered to inittab. admsetup -d is needed to unregister.

[システムの処理]

コマンドは異常終了します。

[対策]

TPBroker が /etc/inittab にすでに登録されているため，再度登録する必要はありません。登録削除するには，admsetup コマンドに -d オプションを指定して入力してください。

KFCB29035-E

TPDIR is not set.

[システムの処理]

コマンドは異常終了します。

[対策]

環境変数 TPDIR を設定してから，再度コマンドを入力してください。

KFCB29037-E

Cannot access ServiceControlManager.

[システムの処理]

コマンドは異常終了します。

[対策]

Administrators 権限のあるユーザで再度コマンドを入力してください。

KFCB29038-E

Failure to open ServiceControlManager.

[システムの処理]

コマンドは異常終了します。

[対策]

Administrators 権限のあるユーザで再度コマンドを入力してください。

KFCB29039-I

Cannot find specified service [aa...aa].

aa...aa：サービス名

[システムの処理]

コマンドは正常終了します。

[対策]

なし。

KFCB29040-E

Failure to open service [aa...aa].

aa...aa：サービス名

[システムの処理]

コマンドは異常終了します。

[対策]

Administrators 権限で実行しているかどうかを確認してください。

KFCB29041-E

Cannot stop specified service [aa...aa].

aa...aa：サービス名

[システムの処理]

コマンドは異常終了します。

[対策]

サービスが動作中です。サービスコントロールパネルからサービスを停止したあと，再度コマンドを入力してください。

KFCB29042-E

The specified service [aa...aa] has not stopped yet.

aa...aa：サービス名

[システムの処理]

コマンドは異常終了します。

[対策]

サービスが動作中です。サービスコントロールパネルからサービスを停止したあと，再度コマンドを入力してください。

KFCB29043-E

The specified service [aa...aa] has been already deleted.

aa...aa：サービス名

[システムの処理]

コマンドは異常終了します。

[対策]

なし。

KFCB29044-E

Failure to delete service [aa...aa].

aa...aa：サービス名

[システムの処理]

コマンドは異常終了します。

[対策]

Administrators 権限で再度コマンドを入力してください。

KFCB29045-E

The specified service [aa...aa] has already existed.

aa...aa：サービス名

[システムの処理]

コマンドは異常終了します。

[対策]

なし。

KFCB29046-E

Cannot create specified service [aa...aa].

aa...aa：サービス名

[システムの処理]

コマンドは異常終了します。

[対策]

admsetup の実行ユーザに Administrators 権限があることを確認してください。

KFCB29047-E

Spool directory exists. Please execute 'admsetup -d'.

[システムの処理]

コマンドは異常終了します。

[対策]

-d オプションを指定した admsetup コマンドを入力してから，再度 -c オプションを指定した admsetup コマンドを入力してください。

KFCB29048-E

Cannot create directory. path=aa...aa.

aa...aa：パス名

[システムの処理]

コマンドは異常終了します。

[対策]

指定されたディレクトリに対するアクセス権限があるかどうかを確認してください。

KFCB29049-E

Cannot remove directory. path=aa...aa.

aa...aa：パス名

[システムの処理]

コマンドは異常終了します。

[対策]

再度 -d オプションを指定した admsetup コマンドを入力してください。

KFCB29050-E

Failure to access file. file=aa...aa.

aa...aa：ファイル名

[システムの処理]

コマンドは異常終了します。

[対策]

指定されたディレクトリに対するアクセス権限があるかどうかを確認してください。

KFCB29052-E

Failure to register daemon to /etc/inittab.

[システムの処理]

コマンドは異常終了します。

[対策]

スーパーユーザで再度コマンドを実行してください。

KFCB29053-E

Failure to unregister daemon from /etc/inittab.

[システムの処理]

コマンドは異常終了します。

[対策]

スーパーユーザで再度コマンドを実行してください。

KFCB29055-E

[aa...aa] is not setup.

aa...aa：ディレクトリ名

[システムの処理]

コマンドは異常終了します。

[対策]

-d オプションを指定した admsetup コマンドを実行したあと，再度 admsetup コマンドを入力してください。

KFCB29056-E

Fatal error occurred.

[システムの処理]

コマンドは異常終了します。

[対策]

保守員に連絡してください。

KFCB29057-E

The admd configuration file is invalid.

[システムの処理]

コマンドは異常終了します。

[対策]

admsetup コマンドで指定されたプロセス監視定義ファイルが存在しません。正しいプロセス監視定義ファイル名を指定して，再度 admsetup コマンドを入力してください。

KFCB29058-E

System definition is not set. Please execute 'tssetup'.

[システムの処理]

コマンドは異常終了します。

[対策]

tssetup コマンドを入力してシステム環境定義を設定してください。

KFCB29059-E

Specified arguments are invalid combination.

[システムの処理]

コマンドは異常終了します。

[対策]

admstartprc または admstopprc コマンドで複数の識別子を指定した場合，-o オプ

ションを指定しないで再度実行してください。

KFCB29060-E

Number of specified IDs exceeded the limit.

［システムの処理］

コマンドは異常終了します。

［対策］

admstartprc または admstopprc コマンドで指定する識別子の数が多過ぎます。64 個以内の識別子を指定してから再度実行してください。

KFCB29061-E

$ADMFS directory exists.

［システムの処理］

コマンドは異常終了します。

［対策］

環境変数 ADMFS で設定されるディレクトリがすでに存在するので，次のどれかの操作をしてください。

1. 環境変数 ADMFS を別のディレクトリに設定して，再度コマンドを実行する。
2. -i オプションを指定して admsetup コマンドを実行する。
3. -i オプションを指定した admsetup コマンドでセットアップした環境を作成し直す場合は，環境変数 ADMFS に設定されているディレクトリを削除してから再度コマンドを実行する。

KFCB29062-E

Cannot share $ADMFS directory.

［システムの処理］

コマンドは異常終了します。

［対策］

環境変数 ADMFS で設定されるディレクトリを共用する場合は，同じバージョンの TPBroker で共用するように環境を設定したあとで，コマンドを再度実行してください。環境変数 ADMFS で設定されるディレクトリを共用する必要がない場合は，-i オプションを指定しないで再度実行してください。

KFCB29063-E

Lack of ID on /etc/inittab.

[システムの処理]

admsetup コマンドは，処理を中断します。

[対策]

OS に登録されている TPBroker 環境の数が 5 を超えています。不要な環境を -d オプションを指定した admsetup コマンドで削除して，再度実行してください。

KFCB29064-E

ADMD downed.

[システムの処理]

コマンドは異常終了します。

[対策]

admlogcat コマンドで，ADM デーモンが異常終了した原因を調べてください。原因は，次のどちらかになります。

1. 監視対象プロセスの起動に失敗した場合の定義が "down" に設定されていて，プロセスの起動に失敗した。
2. メモリ不足が発生した。

KFCB29065-W

Specified process is alive. This change was ignored. ID = aa...aa

aa...aa：識別子

[システムの処理]

aa...aa は起動中のため，プロセス監視定義の変更は行いません。コマンドは処理を続けます。

[対策]

起動中のプロセスに対するプロセス監視定義を変更する場合，次のどちらかの操作をしてください。

1. admstopprc コマンドを実行して該当プロセスを停止させてから，admreload コマンドを実行する。
2. -f オプションを指定した admreload コマンドを実行する。

KFCB29066-W

The TPBroker system environment is intended to be changed. This change was ignored.

[システムの処理]

TPBroker システム全体の環境変数が変更されています。この環境変数は反映されません。コマンドは処理（プロセス監視定義の追加，削除，変更）を続けます。

[対策]

TPBroker システム全体の環境変数を変更する場合は，admstop コマンドを実行し，TPBroker を正常終了させてから，再開始してください。

KFCB29067-I

The definition has not been changed.

[システムの処理]

プロセス監視定義に変更がありませんでした。コマンドは正常終了します。

[対策]

なし。

KFCB29068-W

Number of monitored processes exceeds the value defined in the system definition. (aa...aa)

aa...aa：起動したプロセスの識別子

[システムの処理]

運用定義 /ADM/max_process_num で指定した数以上のプロセスがすでに定義されているか，または起動されています。aa...aa を追加できません。コマンドは処理（プロセス監視定義の追加，削除，変更）を続けます。

[対策]

プロセス監視定義の追加と同時にプロセス監視定義の削除があり，運用定義 /ADM/max_process_num で指定した数より少ないプロセス数となった場合，admreload コマンドを再度入力してください。または，運用定義 /ADM/max_process_num で監視できるプロセス数の最大値を増加させ，admstop コマンドを実行し，TPBroker を正常終了させてから，再開始してください。また，admreload コマンドを実行した場合，プロセス監視定義ファイルが削除されて存在しないときは，メッセージ KFCB29021-E を出力します。それ以外のエラー発生時は，対応したメッセージを出力します。各メッセージに従って対策してください。

KFCB29069-E

Specified service [TPBroker] is marked for delete.

[システムの処理]

admsetup コマンドは異常終了します。

[対策]

［サービス］ウィンドウを閉じて，-d オプションを指定した admsetup コマンドを入力してから再度 -c オプションを指定した admsetup コマンドを実行してください。

KFCB29070-I

ADMD has already terminated.

［システムの処理］

ADM デーモンがすでに終了しているので何もしません。

［対策］

なし。

KFCB29071-I

Process started (ID=aa...aa, pid= bb...bb)

aa...aa：起動したプロセスの識別子

bb...bb：起動したプロセスのプロセス ID

［システムの処理］

プロセスを起動します。

［対策］

なし。

KFCB29072-I

Process stopped (ID=aa...aa, pid= bb...bb)

aa...aa：停止したプロセスの識別子

bb...bb：停止したプロセスのプロセス ID

［システムの処理］

プロセスを停止します。

［対策］

なし。

KFCB29073-W

Cannot remove status file. filename=aa...aa info1=bb...bb info2=cc...cc

aa...aa：TPBroker システムステータスファイル名

bb...bb：保守情報 1

cc...cc：保守情報 2

［システムの処理］

コマンドは処理を続けます。

［対策］

環境変数 ADMFS を使用している TPBroker 環境が起動中でないことを確認してください。または，aa...aa で示されるファイルが使用中でないことを確認してください。確認後，コマンドを再度実行してください。
- ファイルが存在しない場合，メッセージを出力しないで処理を続けます。
- クラスタシステムで環境変数 ADMFS を共有する構成の場合，全ノードの運用支援機能の環境を削除する場合だけ -d オプションおよび -f オプションを指定した admsetup コマンドを実行してください。

KFCB29075-E

[aa...aa] is not registered to service.

aa...aa：サービス名

[システムの処理]

aa...aa はサービスに登録されていません。コマンドは終了します。

[対策]

aa...aa がサービスに登録されているかどうか確認してください。

KFCB29078-E

Length of service name [aa....aa] is over the maximum.

ADM サービス名が最大長を超えています。

aa....aa：TPBroker サービス名称

[システムの処理]

コマンドは異常終了します。

[対策]

サービス名には 1 ～ 32 文字の半角英数字を指定してください。

KFCB29079-E

The path of ADMSPOOL cannot be obtained from the registry.

レジストリから ADMSPOOL のパスが取得出来ません

[システムの処理]

コマンドは異常終了します。

[対策]

レジストリから ADM の SPOOL ディレクトリのパスが取得出来ないため，手動で ADM の SPOOL ディレクトリを削除して ADM のセットアップを再度行ってください。

KFCB29080-E

Specified service name is invalid [aa....aa].

サービス名称が正しくありません。

aa....aa：ADM サービス名

[システムの処理]

コマンドは異常終了します。

[対策]

ADM サービス名に半角英数字 , アンダーバー及び，半角スペース以外の文字が含まれていないか確認してください。また，ADM サービス名に指定した文字列の先頭と語尾に空白文字が含まれていないか，確認した上でセットアップを再度行ってください。

KFCB29082-E

Service = aa....aa,Port = bb....bb,Command = cc....cc,Error code = dd....dd

aa....aa：ADM サービス名

bb....bb: ポート番号

cc....cc: コマンド名

dd....dd: エラーコード（1 ～ 3 の範囲）

[システムの処理]

コマンドは異常終了します。

[対策]

コマンド cc....cc が ADM デーモンにアクセス出来ない時に出力されます。aa....aa の ADM サービス名でコントロールパネルのサービスに登録された TPBroker が bb....bb のポート番号で起動しているか，またはエラーコードに応じて以下の内容から，エラー要因を確認して対処してください。

エラーコード	意味	対策
1	ADM デーモンが未起動であるため，エラーが発生しました。	ADM デーモンが起動しているか，確認してください。ADM デーモンが起動している場合，ADM デーモンの環境変数 ADMSPOOL が，実行したコマンドの環境変数 ADMSPOOL と一致しているか，確認してください。
2	ADM デーモンとの通信に失敗しました。	指定したコマンドの環境変数 TPSPOOL と ADM デーモンの環境変数 TPSPOOL が一致しているか，確認してください。一致している場合，システム定義 /ADM/port_id_info で定義されているポート番号で ADM デーモンが起動しているか，確認してください。

エラーコード	意味	対策
3	致命的なエラーが発生しました。	保守員に連絡してください。

KFCB29083-E

[admsetup-d] was executed to the environment of ADM built by [admsetup -c conffile -r service]. Please specify a [-r service-name] and execute 'admsetup -d'.

ADM サービス名付きでセットアップされた ADMSPOOL ディレクトリを [admsetup -d] で削除しようとしました。

[システムの処理]

コマンドは異常終了します。

[対策]

-r [service-name] を指定して，admsetup を再実行してください。

KFCB29084-E

Cannot create rc script

rc スクリプトの登録に失敗しました。

[システムの処理]

コマンドは異常終了します。

[対策]

root 権限で admsetup コマンドを実行しているか，確認してください。

KFCB29085-E

Cannot delete rc script.

rc スクリプトの削除に失敗しました。

[システムの処理]

コマンドは異常終了します。

[対策]

root 権限で admsetup コマンドを実行しているか，確認してください。

KFCB29086-I

aa...aa was started with bb...bb wait-time.

aa...aa: コマンド名

bb...bb: 指定した ADM デーモン起動待ち時間

[意味]

aa...aa が，bb...bb で示される ADM デーモン起動待ち時間をオプションにして実行されました。

[システムの処理]

コマンドは ADM デーモンの起動を最大 bb...bb 秒待ち合わせます。

[対策]

なし。

KFCB29087-E

Cannot change standard I/O. info1=aa...aa, info2=bb...bb, info3=cc...cc

aa...aa: 保守情報 1

bb...bb: 保守情報 2

cc...cc: 保守情報 3

[意味]

標準入出力の変更ができませんでした。

[システムの処理]

admlaunchux コマンドは異常終了します。

[対策]

/dev/null が存在し，読み取り・書き込み権限があることを確認してください。

KFCB29088-E

Cannnot change signal. signal=aa...aa, info1=bb...bb

aa...aa: 変更に失敗したシグナル値

bb...bb: 保守情報 1

[意味]

シグナルの扱いが変更できませんでした。

[システムの処理]

admlaunchux コマンドは異常終了します。

[対策]

システム上のメモリが破壊されていないことを確認してください。

KFCB29089-E

Cannot execute command. command=aa...aa, reason=bb...bb, info1=cc...cc

aa...aa: 起動に失敗した起動コマンド

bb...bb: 理由コード

access denied: コマンドを起動する権限が不足しています。

not exist: コマンドが存在しません。

invalid format: コマンドのフォーマットが不正です。

unknown: 上記以外の理由です。

cc...cc: 保守情報 1

[意味]

起動コマンドの実行が出来ませんでした。

[システムの処理]

admlaunchux コマンドは異常終了します。

[対策]

理由コードに従って対処してください。

access denied:

起動コマンドに実行権限があるか，起動コマンドのパスを構成するディレクトリにサーチ権限があるかを確認してください。

not exist:

起動コマンドまたは起動コマンドのパスを構成するディレクトリが存在することを確認してください。または絶対パスで起動コマンドを指定していることを確認してください。
または admlaunchux コマンドをプロセス監視定義ファイルに記載した場合に起動コマンドのパスが環境変数を含む記述になっていないことを確認してください。

invalid format:

起動コマンドが壊れていないか確認してください。起動コマンドがシェルスクリプトの場合，先頭の 2 バイトが「#!」であることを確認してください。

unknown:

以下の内容を確認してください。

- 起動コマンドの引数リストに環境変数を加えたバイト数が OS の制限を越えていない。
- 互換のないアーキテクチャの起動コマンドでないか。
- 起動コマンドのパスを構成するディレクトリで，シンボリックリンクが循環していないか。
- 起動コマンドがシステムによって決められた最大値以上のメモリを要求していないか。
- 起動コマンドのパスの構成要素がディレクトリであるか。

KFCB29090-E

System definition is not set. Please execute 'tssetup' or 'tstpbsetup'.

［意味］

システム環境定義が設定されていません。tssetup コマンド，または tstpbsetup コマンドを実行してください。

［システムの処理］

コマンドは異常終了します。

［対策］

tssetup コマンド，または tstpbsetup コマンドを入力してシステム環境定義を設定してください。

KFCB29100-E

Configuration file (aa...aa) does not exist.

aa...aa：プロセス監視定義ファイル名

［システムの処理］

ADM デーモンはプロセス監視定義ファイルが指定されていないものとみなして，処理を続けます。

［対策］

プロセス監視定義ファイル (aa...aa) が見つかりません。正しいプロセス監視定義ファイルを指定して ADM デーモンを再開始してください。プロセス監視定義ファイルは admsetup コマンドでセットアップ時に指定します。このとき，ドライブ名を含め絶対パス名で指定してください。

KFCB29102-E

Failure to allocate memory (size = aa...aa).

aa...aa：確保しようとしたメモリの大きさ

［システムの処理］

ADM デーモンが異常終了します。

［対策］

ほかのアプリケーションプログラムを終了させ，再度 ADM デーモンを起動してください。

KFCB29103-E

Failure to allocate memory (aa...aa).

aa...aa：メモリ不足を起こした関数名

［システムの処理］

ADM デーモンが異常終了します。

［対策］

ほかのアプリケーションプログラムを終了させ，再度 ADM デーモンを起動してください。

KFCB29104-E

Failure to initialize system.

［システムの処理］

ADM デーモンは異常終了します。

［対策］

保守員に連絡してください。

KFCB29105-E

Error is found in the configuration file at aa...aa [bb...bb] cc...cc.

aa...aa：不正な定義値が設定されている識別子（識別子が未設定の場合は，aa...aa には何も表示されません）

bb...bb：カラム番号

cc...cc：次に示す詳細情報

Number of definition is invalid.

1 エントリに対する項目数が不正です。

dd...dd:Error is found.

項目 dd...dd にエラーがあります。

ID(dd...dd) is too long or too short.

識別子 dd...dd の長さが不正です。

ID(dd...dd) is invalid format.

識別子 dd...dd のフォーマットが不正です（英数字以外の文字が使用されています）。

dd...dd is not file.

dd...dd はファイルではありません（ディレクトリが指定されている場合など）。

Command is not defined.

監視対象プロセスが異常終了したときの処理が "command" の場合に起動するコマンドが定義されていません。または，間接起動方式で，プロセス ID 取得用コマンドおよびプロセス停止用コマンドが定義されていません。

dd...dd is invalid.

定義値が不正です（restart，none，command，および down 以外を指定しています。または，設定できる範囲を超えています）。

dd...dd is not a number.

dd...dd は数字ではありません。

dd...dd is too long.

環境変数またはコマンド名／パス名の設定が長過ぎます。

Environment variable is not set.

"PUTENV" で環境変数が設定されていません。

dd...dd is not reserved.

"PUTENV" を定義する個所に dd...dd と記載されています。

Error is found

何らかのエラーが発生しました。または，前のエントリの定義項目数が不正です。

ID (dd...dd) has been already registered.

識別子 dd...dd はすでに登録されています。または，プロセス監視定義ファイル内に同じ識別子が設定されています。

［システムの処理］

ADM デーモンは，不正が見つかったプロセス監視定義を無視し，続きのプロセス監視定義を読み込む処理を続けます。

［対策］

ADM デーモンは異常終了しないので，そのまま処理を続けます。正しいプロセス監視定義を有効にするには，admstop コマンド入力後，次に示す項目を確認してから再度 ADM デーモンを起動してください。

1. 識別子が正しいか。
2. プロセス名が正しいか。
3. 設定値が正しいか。
4. 項目数が正しいか。

KFCB29106-E

Process (aa...aa) is not defined in the configuration file.

aa...aa：指定された識別子

［システムの処理］

ADM デーモンは処理を続けます。

［対策］

admstartprc コマンドの -i オプションで指定する識別子が，プロセス監視定義に存在しません。以下のいずれかの対策を行ってください。

- admstartprc コマンドの -i オプションに指定する識別子を変更してコマンドを再

投入する。
- プロセス監視定義の識別子を修正して，admreload コマンドを投入する。
- プロセス監視定義の識別子を修正して，再度 ADM デーモンを起動する。

KFCB29107-E

File (aa...aa) is not found.

aa...aa：ファイル名

［システムの処理］

ADM デーモンは処理を続けます。

［対策］

プロセス監視定義ファイルに指定されたファイル名やコマンドのオプションで指定したファイル名が正しいかを確認してください。これらのファイル名はすべて絶対パス名で指定してください。

KFCB29108-E

Failure to get system definition (aa...aa).

aa...aa：定義パラメタ

［システムの処理］

ADM デーモンはデフォルト値で処理を続けます。

［対策］

システム環境定義 aa...aa の取得に失敗しました。定義パラメタ aa...aa が正しく設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合は，tsdefvalue コマンドで設定してください。

KFCB29109-E

Failure to open system definition (aa...aa).

aa...aa：定義キー

［システムの処理］

ADM デーモンは異常終了します。

［対策］

定義キーが設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合は，tssetup コマンドを入力して実行環境を初期化してください。

KFCB29110-E

Invalid definition (aa...aa=bb...bb) is found.

aa...aa：定義パラメタ

bb...bb：定義値

［システムの処理］

ADM デーモンはデフォルト値で処理を続けます。

［対策］

システム環境定義 aa...aa の値が不正です。定義パラメタ aa...aa が正しく設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合は，tsdefvalue コマンドで設定してください。

KFCB29111-I

Invalid definition (aa...aa=bb...bb) is found.

aa...aa：定義パラメタ

bb...bb：定義値

［システムの処理］

ADM デーモンはデフォルト値を使用して，処理を続けます。

［対策］

システム環境定義 aa...aa の値が不正です。定義パラメタ aa...aa が正しく設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合は，tsdefvalue コマンドで設定してください。

KFCB29112-E

Failure to initialize status file.

［システムの処理］

ADM デーモンは異常終了します。

［対策］

$ADMSPOOL が存在するかを確認してください。

KFCB29113-E

Failure to write status file.

［システムの処理］

ADM デーモンは異常終了します。

［対策］

$ADMSPOOL が存在するかを確認してください。

KFCB29114-E

Failure to read status file.

［システムの処理］

ADM デーモンは異常終了します。

［対策］

$ADMSPOOL が存在するかを確認してください。

KFCB29115-E

Failure to open status file.

［システムの処理］

ADM デーモンは異常終了します。

［対策］

$ADMSPOOL が存在するかを確認してください。

KFCB29116-I

Recreate status file.

［システムの処理］

ADM デーモンは処理を続けます。

［対策］

$ADMSPOOL が存在するかを確認してください。

KFCB29117-E

Invalid parameter is found.

［システムの処理］

ADM デーモンは異常終了します。

［対策］

保守員に連絡してください。

KFCB29118-E

Failure to start process (ID=aa...aa).

aa...aa：プロセス監視定義ファイルまたは運用コマンドで指定された識別子

[システムの処理]

1. プロセス監視定義ファイルに設定された実行可能ファイルの起動に失敗した場合，定義に従って，次のうちどれか一つの動作をします。
   - ADM デーモンが起動に失敗した実行可能ファイルを再開始します。
   - ADM デーモンが異常終了します。
   - 何もしないで処理を続けます。
2. 運用コマンドで指定された実行可能ファイルの起動に失敗した場合，運用コマンド入力時に指定されたオプションに従って，次のどちらかの動作をします。
   - ADM デーモンが異常終了します。
   - 何もしないで処理を続けます。

[対策]

指定された識別子に対する実行可能ファイルが存在するか，および実行権限が与えられているかどうかを確認してください。ADM デーモンが異常終了した場合，正しいプロセス監視定義ファイルを指定して ADM デーモンを再開始してください。[システムの処理] の 2. の「何もしないで処理を続ける」場合，再度運用コマンドを入力してください。

KFCB29119-E

Failure to create process (ID=aa...aa).

aa...aa：プロセス監視定義ファイルまたは運用コマンドで指定された識別子

[システムの処理]

1. プロセス監視定義ファイルに指定された実行可能ファイルの起動に失敗した場合，定義に従って，次のうちどれか一つの動作をします。
   - ADM デーモンが起動に失敗した実行可能ファイルを再開始します。
   - ADM デーモンが異常終了します。
   - 何もしないで処理を続けます。
2. 運用コマンドで指定された実行可能ファイルの起動に失敗した場合，運用コマンド入力時に指定されたオプションに従って，次のどちらかの動作をします。
   - ADM デーモンが異常終了します。
   - 何もしないで処理を続けます。

[対策]

指定された識別子に対する実行可能ファイルが存在するか，および実行権限が与えられているかどうかを確認してください。ADM デーモンが異常終了した場合，正しいプロセス監視定義ファイルを指定して ADM デーモンを再開始してください。[システムの処理] の 2. の「何もしないで処理を続ける」場合，再度運用コマンドを入力してください。

KFCB29120-E

Failure to get process ID (ID=aa...aa).

aa...aa：プロセス監視定義ファイルまたは運用コマンドで指定された識別子

[システムの処理]

ADM デーモンは，アプリケーションプログラムを監視しません。

[対策]

Windows 版の場合は，16 ビットアプリケーションを監視対象にできません。プロセスが起動しているかを確認してください。

KFCB29121-E

Failure to start watch process (PID=aa...aa).

aa...aa：プロセス ID

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29122-E

Failure to register watch handler.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29123-E

Failure to unregister watch handler.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29124-E

Failure to stop monitoring process (PID=aa...aa).

aa...aa：プロセス ID

［システムの処理］
ADM デーモンは異常終了します。

［対策］
保守員に連絡してください。

KFCB29125-I

ADMD started (PID=aa...aa, TIME=bb...bb).

aa...aa：ADM デーモンのプロセス ID

bb...bb：起動時刻

［システムの処理］
処理を続けます。

［対策］
なし。

KFCB29126-E

Process (PID=aa...aa, ID=bb...bb) downed.

aa...aa：異常終了したプロセスのプロセス ID

bb...bb：異常終了したプロセスに対する識別子

［システムの処理］
監視中のプロセスが異常終了しました。プロセス監視定義に従って処理を進めます。

［対策］
なし。

KFCB29127-I

Restart the down process (ID=aa...aa).

aa...aa：異常終了したプロセスに対する識別子

［システムの処理］
異常終了したプロセスを再起動します。

［対策］
なし。

KFCB29128-I

aa...aa is executed.

aa...aa：プロセス監視定義ファイルに設定されたコマンド

[システムの処理]

プロセス監視定義ファイルに設定されたコマンドを実行します。

[対策]

なし。

KFCB29129-I

ADMD do nothing.

[システムの処理]

何もしません。

[対策]

なし。

KFCB29130-I

ADMD downed.

[システムの処理]

ADM デーモンが終了し，OS から再度 ADM デーモンが起動します。

[対策]

なし。

KFCB29131-I

ADMD started automatically.

[システムの処理]

ADM デーモンは自動起動です。

[対策]

なし。

KFCB29132-I

ADMD started manually.

[システムの処理]

ADM デーモンは手動起動です。

[対策]

なし。

KFCB29133-I

ADMD started with normal mode.

[システムの処理]

ADM デーモンは正常開始モードです。

[対策]

なし。

KFCB29134-I

ADMD started with forcible normal mode.

[システムの処理]

ADM デーモンは強制正常開始モードです。

[対策]

なし。

KFCB29135-I

ADMD started with recovery mode.

[システムの処理]

ADM デーモンは再開始モードです。

[対策]

なし。

KFCB29136-I

ADMD stopped with normal mode.

[システムの処理]

ADM デーモンは正常終了モードです。

[対策]

なし。

KFCB29137-I

ADMD stopped with forcible mode.

[システムの処理]

ADM デーモンは強制正常終了モードです。

[対策]

なし。

KFCB29138-I

Process (PID=aa...aa, ID=bb...bb) started.

aa...aa：起動したプロセスのプロセス ID

bb...bb：起動したプロセスの識別子

［システムの処理］

プロセスを起動します。

［対策］

なし。

KFCB29139-I

Process (PID=aa...aa, ID=bb...bb) stopped.

aa...aa：停止したプロセスのプロセス ID

bb...bb：停止したプロセスの識別子

［システムの処理］

プロセスを停止します。

［対策］

なし。

KFCB29140-I

Starting to monitor process (PID=aa...aa, ID=bb...bb).

aa...aa：監視を開始するプロセスのプロセス ID

bb...bb：監視を開始するプロセスの識別子

［システムの処理］

外部から API を使用して，監視を要求したプロセスの監視を開始します。ID にはシステムでユニークに設定した識別子が設定されます。

［対策］

なし。

KFCB29141-I

Stopping to monitor process (PID=aa...aa, ID=bb...bb).

aa...aa：監視を終了するプロセスのプロセス ID

bb...bb：監視を終了するプロセスの識別子

［システムの処理］

外部から API を使用して，監視を要求したプロセスの監視を終了します。

[対策]

なし。

KFCB29142-E

Failure to read shared memory.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29143-E

Function (aa...aa) returns error code (bb...bb).

aa...aa：異常が発生した関数名

bb...bb：関数が返したエラーコード

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29144-E

Failure to write process status.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29145-E

Failure to write system status.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29146-E

Failure to write status.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29147-E

Failure to initialize TCP.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29148-E

Communication error occurred.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29149-I

Command (aa...aa) is executed.

aa...aa：入力されたコマンド

[システムの処理]

コマンドが入力されました。

[対策]

なし。

KFCB29150-I

API (aa...aa) is executed.

aa...aa：実行された API

[システムの処理]

API が実行されました。

[対策]

なし。

KFCB29151-E

Failure to create message log file.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29152-E

Failure to write message to log file.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29156-E

Failure to create shared memory file.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29157-E

Failure to allocate shared memory file.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29158-E

Failure to open shared memory file.

[システムの処理]
ADM デーモンは異常終了します。

[対策]
保守員に連絡してください。

KFCB29159-E

Number of processes which are monitored exceeded the value defined in the system definition.

[システムの処理]
プロセス監視定義ファイルの読み込み時，またはプロセス起動時に運用定義 /ADM/max_process_num に設定された最大監視対象プロセス数を超えました。ADM デーモンは処理を続けます。

[対策]
運用定義 /ADM/max_process_num を変更後，admstop コマンドで TPBroker を正常終了してから，再度 TPBroker を開始してください。

KFCB29160-I

Abort handler is invoked.

[システムの処理]
監視中のプロセスが異常終了し，ハンドラが起動しました。プロセス監視定義に従って処理します。

[対策]
なし。

KFCB29161-I

System has already invoked.

[システムの処理]
ADM デーモンがすでに起動しているので何もしません。

[対策]
なし。

KFCB29162-I

System has not invoked yet.

[システムの処理]
何もしません。

[対策]

admstart コマンドを入力して，ADM デーモンを使用できる状態にしてください。

KFCB29163-I

ADMD received invalid message format from APIs or command.

[システムの処理]

ADM デーモンは何もしません。

[対策]

なし。

KFCB29164-E

Failure to check previous status file.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29165-E

Failure to create new thread.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29166-E

Failure to execute specified command.

[システムの処理]

ADM デーモンは処理を続けます。

[対策]

指定したコマンドに実行権限があるかどうかを確認してください。

KFCB29167-E

System is invalid status.

[システムの処理]

ADM デーモンは異常終了します。

［対策］

保守員に連絡してください。

KFCB29168-E

Length of file name is too long.

［システムの処理］

プロセス監視定義ファイルやコマンドに指定したファイル名が長過ぎます。ADM デーモンは処理を続けます。

［対策］

適切な長さのファイル名を指定してください。

KFCB29169-E

Process (ID=aa...aa) is not found.

aa...aa：プロセスに対する識別子

［システムの処理］

ADM デーモンは異常終了します。

［対策］

保守員に連絡してください。

KFCB29170-E

Cannot stop the process (ID=aa...aa).

aa...aa：プロセスに対する識別子

［システムの処理］

ADM デーモンは異常終了します。

［対策］

保守員に連絡してください。

KFCB29171-E

Environment is invalid.

［システムの処理］

ADM デーモンは異常終了します。

［対策］

環境変数を見直してください。

KFCB29172-I

The specified process (aa...aa) has been already monitored.

aa...aa：指定された識別子

[システムの処理]

ADM デーモンは処理を続けます。

[対策]

なし。

KFCB29173-I

The specified process (aa...aa) has not been monitored yet.

aa...aa：指定された識別子

[システムの処理]

ADM デーモンは処理を続けます。

[対策]

なし。

KFCB29179-I

The specified port number(aa...aa) is invalid or used. ADMD uses bb...bb .

aa...aa：システム環境定義で設定されたポート番号

bb...bb：自動的に割り当てたポート番号

[システムの処理]

自動的に割り当てたポート番号をシステム環境定義に設定します。ADM デーモンとコマンドとの通信はこのポート番号を使用します。

[対策]

なし。

KFCB29180-W

Number of environment variables in configuration file is over the maximum.

[システムの処理]

ADM デーモンは処理を続けます。

[対策]

プロセス監視定義ファイルに設定できる環境変数の数は 100 以下のため，プロセス監視定義ファイルを確認してください。

KFCB29181-E

$ADMSPOOL or $ADMFS directory does not exist.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

admsetup コマンドを入力し，実行環境を構築してください。

KFCB29182-W

Number of invoked processes indirectly exceeded the limit(100).

[システムの処理]

ADM デーモンは処理を続けます。

[対策]

なし。

KFCB29183-W

Configuration file is not specified.

[システムの処理]

ADM デーモンは処理を続けます。ただし，プロセス監視定義ファイルが設定されていません。

[対策]

いったん TPBroker を終了し，再度セットアップすることをお勧めします。

KFCB29184-E

The administration daemon terminated abnormally ten continuous times.

The administration daemon stops processing.

[システムの処理]

ADM デーモンの起動を停止します。

[対策]

-d オプションを指定した admsetup コマンドで実行環境を削除したあと，再度 admsetup コマンドで実行環境を構築してください。

KFCB29185-E

Cannot create management file.

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29186-W

ADMD terminated process(es) forcibly. ID=aa...aa

aa...aa：強制停止したプロセスに対する識別子

[システムの処理]

停止コマンドの発行後，監視対象プロセスが停止しなかったため，監視対象プロセスを強制的に停止しました。監視対象プロセスは正常に停止していない可能性があります。

[対策]

なし。

KFCB29187-W

Cannot get process IDs by indirect activation. ID=aa...aa COMMAND=bb...bb

aa...aa：プロセスに対する識別子

bb...bb：プロセス ID 取得用コマンド

[システムの処理]

ADM デーモンはプロセスを監視しないで処理を続けます。

[対策]

間接起動方式で起動された監視対象プロセスが起動しているか，またはプロセス ID 取得用コマンドが正しく動作するかを確認してください。監視対象プロセスが起動している場合は，ADM デーモンの監視対象外になっていますので，手動でプロセスを停止させてください。監視対象プロセスが正しく起動するか，またはプロセス ID 取得用コマンドが正常に動作することを確認して，再度 admstartprc コマンドを実行してください。

KFCB29188-E

The monitored process terminated abnormally continuously.

The administration daemon gave up to restart the process (ID=aa...aa).

aa...aa：異常終了したプロセスに対する識別子

[システムの処理]

異常終了したプロセスを再起動しないでそのままにします。

[対策]

admstartprc コマンドで再起動してください。

KFCB29189-W

A definition (aa...aa=bb...bb) is invalid. cc..cc is assumed.

aa...aa：定義パラメタ

bb...bb：定義に設定された値

cc...cc：定義値として仮定される値

[システムの処理]

値として cc...cc が設定されたものと仮定して処理を続行します。

[対策]

定義パラメタ aa...aa に無効な値が指定されています。設定内容を見直し，正しい値を設定してください。

KFCB29190-W

The order is invalid.(ID=aa...aa)

aa...aa：識別子

[システムの処理]

order が指定されていないものと仮定して処理を続行します。

[対策]

プロセス監視定義ファイル中の識別子 aa...aa で設定されている order が不正，または上限を超えています。設定内容を見直してください。

KFCB29191-I

$ADMFS directory was created.

[システムの処理]

$ADMFS ディレクトリがないため，ADM デーモンは $ADMFS ディレクトリを作成し，処理を続行します。

[対策]

-i オプションを指定して admsetup コマンドを実行した場合の初回起動時は対策の必要はありません。それ以外の場合，$ADMFS ディレクトリがマウントされているか，または $ADMFS ディレクトリが削除されていないかどうかを見直してください。

KFCB29195-I

[aa...aa]Version=bb...bb.TPDIR=cc...cc,TPSPOOL=dd...dd,TPFS=ee...ee,ADMSPOOL=ff...ff,ADMFS=gg...gg,VBROKER_ADM=hh...hh,Confile=ii...ii,port=jj...jj

aa...aa：コントロールパネルのサービスに登録されている ADM サービス名。

bb...bb：TPBroker のバージョン

cc...cc：aa...aa の admd が使用している TPDIR の値

dd...dd：aa...aa の admd が使用している TPSPOOL の値

ee...ee：aa...aa の admd が使用している TPFS の値

ff...ff：aa...aa の admd が使用している ADMSPOOL の値

gg...gg：aa...aa の admd が使用している ADMFS の値

hh...hh：aa...aa の admd が定義している VBROKER_ADM の値

ii...ii：aa...aa の admd が使用している adm の定義ファイル

jj...jj：aa...aa の admd が使用しているポート番号

［システムの処理］

なし

［対策］

なし

KFCB29196-I

The system definition /ADM/set_conf_mode is aa..aa.

ADM の開始モードは aa...aa です。

aa...aa：システム環境定義値（AUTO/MANUAL/MANUAL2/MANUAL3/MANUAL4）

［システムの処理］

なし

［対策］

なし

KFCB29197-I

Prior termination of ADM daemon was due to aa...aa.

前回，ADM デーモンは理由コード aa...aa により，終了しました。

admsetup 後に初めて開始したときにも出力します。

aa...aa：理由コード

理由コード	意味
1	admstop コマンドによる停止
2	OS ダウンによる停止
3	OS シャットダウンによる停止
4	ADM デーモンダウン
5	プロセス監視定義に従った ADM デーモンダウン

[システムの処理]

なし

[対策]

なし

KFCB29198-W

Monitoring process was already downing(PID=aa...aa, ID=bb...bb).

aa...aa: プロセス ID

bb...bb: プロセス監視定義または運用コマンドで指定された識別子

[意味]

監視プロセスは既にダウンしていました。監視プロセスは「ADM デーモンが定義ファイルに従って，監視プロセスを停止させようとするまでの間に」ダウンしています。

[システムの処理]

何もしません。

[対策]

監視プロセスがダウンした原因を調査してください。

KFCB29199-W

Start command of monitoring process(ID=aa...aa) is terminated by ADMD.

aa...aa: プロセス監視定義または運用コマンドで指定された識別子

[意味]

プロセス起動用コマンドが ADM デーモンによって，停止されます。プロセス起動用コマンドのタイムアウト値を過ぎても，プロセス起動用コマンドが終了しませんでした。

[システムの処理]

プロセス起動用コマンドを停止した後，処理を続行します。

[対策]

開始コマンドの終了時間の長さが不当に長い場合，その原因を調査してください。そうでない場合にはプロセス起動用コマンドのタイムアウト値を適切な値に設定した後に admreload コマンドを実行，又は ADM を正常開始で再起動してください。プロセス起動用コマンドのタイムアウト値はプロセス監視定義ファイルにて設定可能です。

KFCB29200-E

Fatal error. info1=aa...aa info2=bb...bb

aa...aa：保守情報 1

bb...bb：保守情報 2

[システムの処理]

コマンドは異常終了します。

[対策]

保守員に連絡してください。

KFCB29201-E

aa...aa: bb...bb

aa...aa：OS の関数名

bb...bb：エラーコード

[システムの処理]

コマンドは異常終了します。

[対策]

このメッセージのあとに続くメッセージを基に対策してください。

KFCB29202-E

Internal error. info1=aa...aa info2=bb...bb

aa...aa：保守情報 1

bb...bb：保守情報 2

[システムの処理]

コマンドは異常終了します。

[対策]

保守員に連絡してください。

KFCB29203-E

aa...aa has failed. reason=bb...bb info1=cc...cc info2=dd...dd

aa...aa：コマンド名

bb...bb：詳細情報

no memory：メモリが不足しています。

environment：環境設定が不足しています。

cc...cc：保守情報 1

dd...dd：保守情報 2

［システムの処理］

コマンドは異常終了します。

［対策］

理由を確認して，設定を見直してください。メモリ不足の場合，使用できるメモリ領域を増やしてください。環境設定不足の場合，実行環境を正しく設定してください。

KFCB29204-E

Cannot start command process. ID=aa...aa cmd=bb...bb

aa...aa：識別子

bb...bb：コマンド列（512 文字以下）

［システムの処理］

コマンドは異常終了します。

［対策］

識別子に対応したコマンド列の実行に失敗しました。コマンド列が正しいかどうかを確認してください。

KFCB29205-E

Cannot read file. ID= aa...aa file=bb...bb

aa...aa：識別子

bb...bb：ファイル名

［システムの処理］

コマンドは異常終了します。

［対策］

ファイルを読み込めません。ファイル名が正しいかどうかを確認してください。

KFCB29206-E

ID conflicts. ID=aa...aa info1=bb...bb info2=cc...cc

aa...aa：識別子

bb...bb：保守情報 1

cc...cc：保守情報 2

［システムの処理］

コマンドは異常終了します。

［対策］

指定された識別子が重複しています。admlaunch コマンドの識別子に正しい値を設定してください。

KFCB29207-E

aa...aa process to be stopped not found. ID=bb...bb

aa...aa：コマンド名

bb...bb：識別子

［システムの処理］

コマンドは異常終了します。

［対策］

指定された識別子で起動中の監視対象プロセスが存在しません。admlaunch コマンドの識別子に正しい値を設定してください。

KFCB29208-I

aa...aa

aa...aa：コマンドの使用方法

［システムの処理］

コマンドは異常終了します。

［対策］

コマンドの使用方法に従って，正しい使用方法で再度コマンドを入力してください。

KFCB29209-E

Cannot create logfile. file=aa...aa

aa...aa：ファイル名

［システムの処理］

コマンドは異常終了します。

[対策]

ログファイルを作成できません。メッセージに出力されたファイルのアクセス権限を確認してください。または，ディスクの空き領域を確保してください。

KFCB29210-E

Illegal argument. reason=aa...aa arg=bb...bb

aa...aa：詳細情報

invalid ID：不正な ID が指定されています。

bb...bb：識別子

[システムの処理]

コマンドは異常終了します。

[対策]

メッセージに従って，不正な引数を見直してください。

KFCB29211-W

Excess characters are ignored. ID=aa...aa file=bb...bb

aa...aa：識別子

bb...bb：起動コマンド列ファイル名

[システムの処理]

コマンドは処理を続けます。

[対策]

コマンド列の長さが最大値を超えています。コマンド列の長さを確認してください。

KFCB29220-W

Stop command of monitoring process(ID=aa...aa) is terminated by ADMD.

aa...aa: プロセス監視定義または運用コマンドで指定された識別子

[意味]

プロセス停止用コマンドが ADM デーモンによって，停止されます。プロセス停止用コマンドのタイムアウト値（300 秒）を過ぎても，プロセス停止用コマンドが終了しませんでした。

[システムの処理]

ADM デーモンはプロセス停止用コマンドを停止した後，処理を続行します。

[対策]

終了コマンドの終了時間が長い場合，その原因を調査してください。

KFCB29221-W

Command for process ID acquisition(ID= aa...aa) is terminated by ADMD.

aa...aa: プロセス監視定義または運用コマンドで指定された識別子

[意味]

プロセス ID を取得するためのコマンドが ADM デーモンによって，停止されます。監視プロセスのプロセス ID を取得するコマンドがタイムアウト値（300 秒）を過ぎても，終了しませんでした。

[システムの処理]

ADM デーモンはプロセス ID を取得するためのコマンドを停止した後，定義ファイルに従って処理を続けます。

[対策]

プロセス ID を取得するコマンドの終了時間が長い原因を調査してください。

KFCB29222-I

Command for process ID acquisition was not specified, so ADMD does not monitor the process(ID=aa...aa).

aa...aa: プロセス監視定義または運用コマンドで指定された識別子

[意味]

プロセスを ID を取得するコマンドが指定されていないため，ADM デーモンはプロセスを監視しません。

[システムの処理]

ADM デーモンはプロセスを監視しません。

[対策]

ADM の定義ファイルに監視プロセスのプロセス ID を取得するコマンドを指定してください。

KFCB29223-I

ADMD is terminating process that was monitoring previously(PID=aa...aa,ID=bb...bb).

aa...aa: プロセス ID

bb...bb: プロセス監視定義または運用コマンドで指定された識別子

[意味]

ADM デーモンは前回，起動中に監視していたプロセスを停止しています。

[システムの処理]

ADM デーモンは前回，起動中に監視していたプロセスをシステムコールで停止します。

[対策]

なし。

KFCB29224-E

Fatal error occurred (line=aa...aa,info1=bb...bb,info2=cc...cc).

aa...aa: 保守情報 1

bb...bb: 保守情報 2

cc...cc: 保守情報 3

[意味]

致命的なエラーが発生しました。

[システムの処理]

ADM デーモンは異常終了します。

[対策]

保守員に連絡してください。

KFCB29225-E

ADMD gave up to restart the process (ID=aa...aa). Monitoring process was already downing bb...bb times within ten minutes.

aa...aa: 指定された識別子

bb...bb: 監視プロセスが異常終了した回数

[意味]

ADM デーモンはプロセスを再起動するのを中止しました。監視プロセスは既に 10 分以内に（bb...bb）回ダウンしていました。

[システムの処理]

コマンドは異常終了して，ADM デーモンは処理を続行します。

[対策]

監視プロセスがダウンした原因を調査してください。監視プロセスをすぐに再起動させるには -c オプションを指定せず，admstartprc を実行して下さい。

KFCB29226-I

The monitored process information. ID=aa...aa DOWN-COUNT=bb...bb DOWN-RETRY-INTERVAL=cc...cc

aa...aa：異常終了したプロセスに対する識別子

bb...bb：連続して異常終了する回数の上限の設定値

cc...cc：監視時間の設定値

［意味］

再起動を抑止したプロセスの連続して異常終了する回数の上限の設定，および監視時間に関する設定値を表示します。

［システムの処理］

KFCB29188-E のシステムの処理に従います。

［対策］

KFCB29188-E の対策に従ってください。

KFCB29500-E

aa...aa returns bb...bb.

aa...aa：異常を検出した関数名

bb...bb：関数のエラーコード

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29501-E

aa...aa: bb...bb

aa...aa：関数名

bb...bb：保守情報

［システムの処理］

プロセスは異常終了します。

［対策］

前後のメッセージを基に対策してください。

KFCB29502-E

Invalid port number(aa...aa) is specified.

aa...aa：指定されたポート番号

［システムの処理］

プロセスは，このメッセージのあとに出力されるメッセージに従って処理を進めます。

［対策］

このメッセージのあとに出力されるメッセージに従ってください。

KFCB29503-W

Not enough memory.

[システムの処理]
プロセスは異常終了します。

[対策]
メモリが不足しています。ほかのアプリケーションプログラムを終了させてください。

KFCB29504-W

Unexpected data was received. data: aa...aa

aa...aa：保守資料

[意味]
想定外のデータを受け取りました。ADM のコマンドが異常終了したか，または不正なアクセスがあった可能性があります。

[システムの処理]
何もしません。

[対策]
ADM のコマンドが異常終了していないか確認してください。異常終了していない場合は不正アクセスがなかったか確認してください。

KFCB29505-E

aa...aa is not recognized as hostname or ip address.

aa...aa：指定されたホスト名または IP アドレス

[システムの処理]
プロセスは，このメッセージのあとに出力されるメッセージに従って処理を進めます。

[対策]
このメッセージのあとに出力されるメッセージに従ってください。

KFCB29506-E

Cannot create or open file. filename: aa...aa

aa...aa：ファイル名

[システムの処理]
プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29507-E

Cannot create temporary file.

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29508-E

Cannot close file. filename: aa...aa

aa...aa：ファイル名

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29509-E

Cannot read data. filename: aa...aa

aa...aa：ファイル名

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29510-E

Cannot write data. filename: aa...aa

aa...aa：ファイル名

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29511-E

Cannot seek file. filename: aa...aa

aa...aa：ファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29512-E

Cannot release file lock.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29513-E

Cannot create directory. directory_name: aa...aa

aa...aa：ディレクトリ名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29514-E

Cannot remove file. filename: aa...aa

aa...aa：ファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29515-E

Cannot rename from aa...aa to bb...bb.

aa...aa：変更前のファイル名またはディレクトリ名

bb...bb：変更後のファイル名またはディレクトリ名

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29516-E

Cannot copy from aa...aa to bb...bb.

aa...aa：コピー元のファイル名またはディレクトリ名

bb...bb：コピー先のファイル名またはディレクトリ名

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29517-W

Cannot change owner of process to aa...aa.

aa...aa：ユーザ ID

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29518-W

Cannot change group of process to aa...aa.

aa...aa：グループ ID

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB29519-W

Cannot set environment variable aa...aa.

aa...aa：環境変数名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29520-E

Cannot create process. filename: aa...aa

aa...aa：ファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29521-W

Cannot get environment variable aa...aa.

aa...aa：環境変数名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29522-E

Cannot create shared memory file. filename: aa...aa

aa...aa：共用メモリファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29523-E

Cannot open shared memory file. filename: aa...aa

aa...aa：共用メモリファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

tssetup コマンドを実行しているか，および環境変数 TPSPOOL に正しい値が設定されているかどうかを確認してください。

KFCB29524-E

Cannot remove shared memory file. filename: aa...aa

aa...aa：共用メモリファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29525-E

Cannot create trace file. filename: aa...aa

aa...aa：トレースファイル名

[システムの処理]

プロセスは処理を続けます。

[対策]

トレースファイルの書き込みに失敗しました。次の項目を確認してください。

- 環境変数 TPSPOOL，ADMSPOOL に設定したディスクの空き容量。
- 実行環境の環境変数 TPSPOOL で tssetup コマンドが実行されているかどうか。
- $TPSPOOL および $ADMSPOOL ディレクトリが作成されているかどうか。
- $TPSPOOL および $ADMSPOOL ディレクトリに書き込み権限があるかどうか。

KFCB29526-E

Cannot open trace file. filename: aa...aa#

aa...aa：トレースファイル名

[システムの処理]

プロセスは処理を続けます。

[対策]

$ADMSPOOL に読み込み権限があるかを確認してください。

KFCB29527-E

Process aborted. PID=aa...aa

aa...aa：プロセス ID

[システムの処理]

プロセスは異常終了します。

[対策]

このメッセージの直前のメッセージを基に対策してください。このメッセージの前にほかのメッセージを出力していない場合は，保守員に連絡してください。

KFCB29528-E

Internal error occurred. Number of process handler is over the maximum.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29529-E

Internal error occurred. Number of wait-object is over the maximum.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29530-E

Internal error occurred. Number of handler-object is over the maximum.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29531-W

Cannot output a message. ID=aa...aa

aa...aa：メッセージ ID

[システムの処理]

メッセージの出力に失敗しました。プロセスは処理を続けます。

［対策］

該当プロセスがオープンできるファイル数を見直してください。または，TPBroker で必要な環境変数が正しく設定されているかどうかを確認してください。

KFCB29532-E

Cannot change owner of process to root.

［システムの処理］

プロセスは異常終了します。

［対策］

$TPDIR/bin/otsd および $TPDIR/bin/tsstop のファイル所有者が root 権限を持つユーザで，アクセスパーミッションが r-sr-xr-x であることを確認してください。

KFCB29533-E

Cannot kill process. pid=aa...aa

aa...aa：プロセス ID

［システムの処理］

プロセスは処理を続けます。該当プロセスでトランザクションタイムアウトが発生した場合，無視されます。

［対策］

$TPDIR/bin/otsd および $TPDIR/bin/tsstop のファイル所有者が root 権限を持つユーザで，アクセスパーミッションが r-sr-xr-x であることを確認してください。

KFCB29534-W

Cannot acquire lock.

［システムの処理］

処理を打ち切ります。

［対策］

このメッセージの直後のメッセージを基に対策してください。

KFCB29535-W

Unable to allocate aa...aa bytes of shared memory.

aa...aa：サイズ

［システムの処理］

プロセスは処理を続けます。

[対策]

OS の限界値を超えています。システム環境定義を変更してください。ご使用の OS が AIX で，次の条件に該当する場合，ヒープサイズを小さくしてください。

- 環境変数 LDR_CNTRL に MAXDATA を設定している場合
- リンケージオプションに -bmaxdata を指定している場合

KFCB29600-E

aa...aa returns bb...bb.

aa...aa：異常を検知した関数名

bb...bb：エラーコード

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29601-E

aa...aa: bb...bb

aa...aa：異常を検知した関数名

bb...bb：保守情報

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29602-E

Internal error occurred. Number of process handler is over the maximum.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29603-E

Internal error occurred. Number of processes which are monitored is over the maximum.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29604-E

Cannot create or open file. filename: aa...aa

aa...aa：ファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29605-E

Process aborted. PID=aa...aa

aa...aa：プロセス ID

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB29606-E

Cannot start system monitoring. rc=aa...aa

aa...aa：エラーコード

[システムの処理]

プロセスは異常終了します。

[対策]

トランザクションサービスがオンライン状態であるか，または OSAgent が動作しているかどうかを確認してください。

KFCB29607-E

Cannot stop system monitoring. rc=aa...aa

aa...aa：エラーコード

[システムの処理]

プロセスは異常終了します。

［対策］

トランザクションサービスがオンライン状態であるか，または OSAgent が動作しているかどうかを確認してください。

KFCB29608-E

Failure to create the status file. reason=aa...aa info1=bb...bb info2=cc...cc

aa...aa：理由コード

bb...bb：保守情報 1

cc...cc：保守情報 2

［システムの処理］

処理を打ち切ります。また，プロセスは異常終了します。

［対策］

理由コードに応じて，次に示す内容に従って対策してください。

理由コード	意味	対策
1	メモリが不足しています。	使用できるメモリ領域を増やしてください。
3	ファイルの作成に失敗しました。	次の内容を見直してください。 • 環境変数 TPFS または ADMFS で設定したディレクトリおよびキャラクタ型スペシャルファイルのアクセス権限。 • 必要なディスクの空き領域。 • ファイルが使用中でないか。 • AIX 版の場合，TPBroker ファイルシステムを初期化した TPBroker のバージョンと，使用している TPBroker のバージョンに不整合がないか。
16	ファイルシステムの状態取得に失敗しました。	次の内容を見直してください。 • 環境変数 TPFS または ADMFS にキャラクタ型スペシャルファイルを設定している場合，tsmkfs コマンドで TPBroker ファイルシステムを初期化したかどうか。 • AIX 版の場合，TPBroker ファイルシステムを初期化した TPBroker のバージョンと，使用している TPBroker のバージョンに不整合がないか。
上記以外	上記以外のエラー	保守員に連絡してください。

KFCB29609-E

Failure to open the status file. reason=aa...aa info1=bb...bb info2=cc...cc

aa...aa：理由コード

bb...bb：保守情報 1

cc...cc：保守情報 2

［システムの処理］

処理を打ち切ります。また，プロセスは異常終了します。

［対策］

理由コードに応じて，次に示す内容に従って対策してください。

理由コード	意味	対策
1	メモリが不足しています。	使用できるメモリ領域を増やしてください。
2	ファイルのオープンに失敗しました。	環境変数 TPFS で設定したディレクトリおよびキャラクタ型スペシャルファイルのアクセス権限を見直してください。または，必要なディスクの空き領域を確保してください。 TPBroker ファイルシステムの初期設定を行っていない場合は，tsmkfs コマンドを実行してください。
上記以外	上記以外のエラー	保守員に連絡してください。

KFCB29610-E

Failure to set the record count of a status group. reason=aa...aa info1=bb...bb info2=cc...cc

aa...aa：理由コード

bb...bb：保守情報 1

cc...cc：保守情報 2

［システムの処理］

処理を打ち切ります。また，プロセスは異常終了します。

［対策］

理由コードに応じて，次に示す内容に従って対策してください。

理由コード	意味	対策
1	メモリが不足しています。	使用できるメモリ領域を増やしてください。
2	ファイルのオープンに失敗しました。	環境変数 TPFS で設定したディレクトリおよびキャラクタ型スペシャルファイルのアクセス権限を見直してください。または，必要なディスクの空き領域を確保してください。
上記以外	上記以外のエラー	保守員に連絡してください。

KFCB29611-E

Failure to remove the status file. reason=aa...aa info1=bb...bb info2=cc...cc

aa...aa：理由コード

bb...bb：保守情報 1

cc...cc：保守情報 2

[システムの処理]

処理を打ち切ります。また，プロセスは異常終了します。

[対策]

理由コードに応じて，次に示す内容に従って対策してください。

理由コード	意味	対策
1	メモリが不足しています。	使用できるメモリ領域を増やしてください。
上記以外	上記以外のエラー	保守員に連絡してください。

KFCB29614-W

Number of processes which are monitored is over the maximum.

[システムの処理]

処理を続けます。

[対策]

このメッセージに続いて KFCB32001-E，KFCB32015-E が出力される場合，トランザクション定義 /OTS/max_process_monitor_count の設定値が不足しています。この定義の設定値を見直し，増やしてください。

KFCB30000-E

Internal error occurred. info1=aa...aa info2=bb...bb

aa...aa：保守情報 1

bb...bb：保守情報 2

[システムの処理]

プロセスを停止します。

[対策]

保守員に連絡してください。

KFCB30001-E

Invalid sequence. info1=aa...aa info2=bb...bb

aa...aa：保守情報 1

bb...bb：保守情報 2

[システムの処理]

プロセスを停止します。

[対策]

保守員に連絡してください。

KFCB30200-E

Internal error occurred. Process has aborted. info1=aa...aa info2=bb...bb

aa...aa：保守情報 1

bb...bb：保守情報 2

［システムの処理］

プロセスを停止します。

［対策］

保守員に連絡してください。

KFCB30400-W

Can not create API trace file. filename:aa...aa reason=bb...bb info=cc...cc code=dd...dd

aa...aa：異常のあったファイル名

bb...bb：理由コード

1：ファイルを開けません

3：ディレクトリがファイルとして指定されています

4：テンポラリファイルが指定されています

cc...cc：保守情報 1

dd...dd：保守情報 2

［システムの処理］

プロセスは異常のあったファイル操作を無視し，回復したあと，処理を続けます。

［対策］

理由コードを参考に，ファイル名に示されたファイルを確認してください。

KFCB30401-W

An error occurred while handling API trace file. info1=aa...aa info2=bb...bb code=cc...cc

aa...aa：理由コード

3：プロセス名を取得できません

6：API トレースファイル数が最大値を超えました

7：メモリが不足しています

8：NULL ポインタが引数として指定されました

10：パスを格納できません

11：パス文字列を格納できません

bb...bb：保守情報 1

cc...cc：保守情報 2

［システムの処理］

プロセスは異常のあった操作を無視し，回復したあと，処理を続けます。

［対策］

info1 = 3 の場合，UAP のプロセス名を短くしてください。
info1 = 6 の場合，スレッド数が少なくなった時点で自動的に回復します。
そのほかの場合，メモリの設定を見直してください。

KFCB30402-I

There is no information in this file.

［システムの処理］

tsedapt コマンドは処理を終了します。

［対策］

なし。

KFCB30403-E

This is an invalid API trace file. filename:aa...aa reason=bb...bb info=cc...cc code=dd...dd

aa...aa：異常のあったファイル名

bb...bb：理由コード

0：ファイルが壊れています
1：ファイルを開けません
2：ファイルを読めません
3：ディレクトリがファイルとして指定されています
4：テンポラリファイルが指定されています

cc...cc：保守情報 1

dd...dd：保守情報 2

［システムの処理］

tsedapt コマンドは処理を終了します。

［対策］

理由コードを参考に，ファイル名に示されたファイルを確認してください。

KFCB30404-E

An error occurred while handling API trace file. info1=aa...aa info2=bb...bb code=cc...cc

aa...aa：理由コード

7：メモリが足りません

9：ファイル上でシークできません

10：パスを格納できません

11：パス文字列を格納できません

bb...bb：保守情報 1

cc...cc：保守情報 2

［システムの処理］

tsedapt コマンドは処理を終了します。

［対策］

エラー原因を調査し，対策したあと，コマンドを再度入力してください。

KFCB30600-W

An internal event happened. info1=aa...aa info2=bb...bb code1=cc...cc code2=dd...dd

aa...aa：保守情報 1

bb...bb：保守情報 2

cc...cc：保守情報 3

dd...dd：保守情報 4

［システムの処理］

プロセスは異常のあった操作を無視し，回復したあと，処理を続けます。

［対策］

保守員に連絡してください。

KFCB30700-E

(aa...aa) bb...bb(cc...cc) failed : dd...dd.

aa...aa：プロセス ID

bb...bb：システムコール名（最大 15 文字の半角英数字）

cc...cc：システムコールを呼び出したモジュール，関数名やシステムコールへの引数の内容など，任意の情報（最大 63 文字の半角英数字）

dd...dd：システムコールエラー時のエラー番号

［システムの処理］

障害の重要度によって，次のどれか一つを実行します。

- 処理を打ち切り，プロセスを異常終了させます。
- 処理を打ち切り，実行中サービスの呼び出し元へリターンします。
- そのまま処理を続けます。

[対策]

TPBroker 内で発行したシステムコールで，エラーが発生しました。次の表に示す内容に従って対策してください。コアファイルが出力されている場合，そのファイルを保存して，保守員に連絡してください。

表 11-1　エラー番号の意味と対策

エラー番号	意味	対策
EMFILE	該当プロセスでオープンしたファイル数が，OS で規定されている上限値を超えました。	OS の定義を見直し，必要に応じて変更したあと，OS の定義を再作成してください。
ENFILE	ノード上にあるプロセスからのオープン要求が，OS で規定されている最大値を超えました。	
ENOLCK	ノード上にあるプロセスからのファイルロック要求が，OS で規定されている最大値を超えました。	
ENXIO	キャラクタ型スペシャルファイルに対応するデバイスが存在しません。	TPFS または ADMFS 環境変数が共用ディスク上のキャラクタスペシャルファイルの場合は，共用ディスクが活性化していることを確認してください。
上記以外	上記以外のエラー	システムコール名とエラー番号を基に，該当するマニュアルで原因を調査し，アプリケーションプログラムの修正，システム環境定義の変更，または OS の定義の再作成をしてください。

KFCB30708-E

TPBroker file system area aa...aa cannot be initialized, because of being used by other process.

aa...aa：fcntl システムコールによってロックされている TPBroker ファイルシステム領域名

[システムの処理]

処理を打ち切ります。

[対策]

fcntl システムコールによってロックされているプロセスを停止させたあと，コマンドを実行してください。

KFCB30709-E

Cannot initialize TPBroker file system area aa...aa because of memory shortage.

aa...aa：TPBroker ファイルシステム領域名

[システムの処理]

処理を打ち切ります。

[対策]

次の項目について検討したあと，再度実行してください。

- 初期化する容量を減らす。
- 容量を満たすスペシャルファイルを指定する。
- ファイルシステム中にあるほかのファイルを削除する。

KFCB30710-E

Only superuser and the owner of TPBroker file system area can execute this command.

[システムの処理]

実行中の処理を中断します。

[対策]

このコマンドはスーパーユーザ，または TPBroker ファイルシステム領域の所有者が実行してください。

KFCB30711-E

TPBroker file aa...aa is not found.

aa...aa：ユーザが指定した TPBroker ファイル名

[システムの処理]

実行中の処理を終了します。

[対策]

該当する TPBroker ファイルがないか，または指定したファイル名が不正なため，検索できません。正しい TPBroker ファイル名を指定して，再度実行してください。

KFCB30712-E

Invalid argument for option flag aa...aa specified with command bb...bb.

aa...aa：ユーザが指定したフラグ引数

bb...bb：ユーザが指定したコマンド名

[システムの処理]

実行中の処理を打ち切ります。

[対策]

このメッセージの直後に出力される使用方法のメッセージに従って，再度入力してください。

KFCB30714-E

Command argument is invalid.

[システムの処理]
コマンドは異常終了します。

[対策]
コマンド引数の指定がないか，または指定できるコマンド引数の個数よりも多くのコマンド引数が指定されています。このメッセージの直後に出力される使用方法のメッセージに従って，再度入力してください。

KFCB30715-E

Lack of mandatory option flag, or invalid combination of option flags.

[システムの処理]
コマンドは異常終了します。

[対策]
必須のオプションフラグが指定されていないか，またはオプションフラグの組み合わせが不正です。このメッセージの直後に出力される使用方法のメッセージに従って，再度入力してください。

KFCB30716-E

This command cannot handle version aa...aa of the TPBroker file system area (bb...bb).

aa...aa：TPBroker ファイルシステム領域のバージョン番号

bb...bb：TPBroker ファイルシステム領域名

[システムの処理]
処理を打ち切ります。

[対策]
保守員に連絡してください。

KFCB30717-E

No access authority for TPBroker file system area aa...aa.

aa...aa：ユーザが指定した TPBroker ファイルシステム領域名

[システムの処理]
処理を打ち切ります。

[対策]
そのファイルのアクセスモードを変更するか，またはアクセス権限のあるユーザで

再度実行してください。

KFCB30718-E

Number of open files in TPBroker file system area aa...aa exceeds the upper limit.

aa...aa：ユーザが指定した TPBroker ファイルシステム領域名

[システムの処理]

処理を打ち切ります。

[対策]

TPBroker ファイルシステム領域 (aa...aa) のオープン処理で上限値オーバが報告されました。正しい値を指定するか，または不要にオープンしているファイルをクローズしたあとで，再度実行してください。

KFCB30719-E

TPBroker file system area aa...aa is not found.

aa...aa：ユーザが指定した TPBroker ファイルシステム領域名

[システムの処理]

処理を打ち切ります。

[対策]

正しい TPBroker ファイルシステム領域名を指定して，再度実行してください。

KFCB30720-E

Length of TPBroker file system area name aa...aa is invalid.

aa...aa：TPBroker ファイルシステム領域名

[システムの処理]

処理を打ち切ります。

[対策]

指定した TPBroker ファイルシステム領域名が 49 文字を超えています。適切な TPBroker ファイルシステム領域名を指定して，再度実行してください。

KFCB30721-E

TPBroker file system cannot be built in file aa...aa.

aa...aa：ユーザが指定したキャラクタ型スペシャルファイル名

[システムの処理]

処理を打ち切ります。

［対策］

指定したファイルがキャラクタ型スペシャルファイルでないか，またはファイルに対応する装置がありません。正しいスペシャルファイル名を指定して，再度実行してください。

KFCB30722-E

File aa...aa is not a TPBroker file system.

aa...aa：ユーザが指定したキャラクタ型スペシャルファイル名

［システムの処理］

処理を打ち切ります。

［対策］

正しいファイル名を指定して，再度実行してください。

KFCB30723-E

Failed to lock the TPBroker file system.

［システムの処理］

実行中の処理を中断します。

［対策］

ファイルシステムのロック処理で使用する fcntl システムコールで，ロックの上限値オーバが発生しました。ロックできるレコード数のシステム定数を変更して，システムを再構築してください。

KFCB30726-E

Failed to allocate process-specific memory.

［システムの処理］

処理を打ち切ります。

［対策］

不要なプロセスを削除して，再度実行してください。

KFCB30727-E

I/O error occurred in TPBroker file system area aa...aa.

aa...aa：TPBroker ファイルシステムがある TPBroker ファイルシステム領域名

［システムの処理］

管理領域を片面に切り替えます。両面が入出力エラーの場合は，実行中の処理を中断します。

[対策]

TPBroker ファイルシステムの管理領域で入出力エラーが発生しました。TPBroker ファイルシステム領域に割り当てたディスクパーティションを見直してください。

KFCB30728-E

Error occurred during TPBroker file service. Maintenance info:aa...aa:bb...bb.

aa...aa：保守情報 1

bb...bb：保守情報 2

[システムの処理]

TPBroker ファイルサービスの処理を打ち切ります。

[対策]

TPBroker ファイルサービスで異常が発生しました。このメッセージの内容を記録し，保守員に連絡してください。

KFCB30800-E

aa...aa(bb...bb) is failed.errno=cc...cc

aa...aa：エラーとなったシステムコール

bb...bb：内部情報（システムコールの呼び出し元モジュール名）

cc...cc：システムコールのエラー番号

[システムの処理]

システムコールでエラーが発生しました。障害の重要度によって，次のどれかの処置をします。

- 処理を打ち切り，プロセスを異常終了させます。
- 処理を打ち切り，実行中サービスの呼び出し元へリターンします。
- 処理を続けます。

[対策]

エラー番号を参照して，原因を確認して対策してください。

KFCB30801-E

aa...aa(bb...bb) killed by code=cc...cc

aa...aa：エラーを検出した内部処理コード

bb...bb：停止したプロセスのプロセス ID

cc...cc：アボートコード（異常終了要因コード）

[システムの処理]

異常が発生したため，プロセスを停止します。

［対策］

異常終了の原因を調査し，プロセスを再起動してください。

KFCB30900-E

"aa...aa(bb...bb)" failed.errno=cc...cc

aa...aa：エラーとなったシステムコール

bb...bb：内部情報（システムコールの呼び出し元ファイル名）

cc...cc：システムコールのエラー番号

［システムの処理］

システムコールでエラーが発生しました。処理を中止します。

［対策］

システムコールがエラーとなった原因を調査してください。

KFCB31000-E

"TPDIR" is not set.

［システムの処理］

ADM デーモン，OTS デーモン，およびコマンドが終了します。

［対策］

環境変数 TPDIR が未設定です。環境変数 TPDIR を設定してください。

KFCB31001-E

$TPSPOOL directory does not exist.

［システムの処理］

OTS デーモン，コマンドが終了します。

［対策］

環境変数 TPSPOOL の値が正しく設定されているかどうかを確認してください。

tssetup コマンドを実行して，TPBroker の実行環境を作成してください。

KFCB31002-I

tssetup successful.

tssetup **コマンドまたは** tstpbsetup **コマンドは正常終了しました。**

［システムの処理］

コマンドは正常終了します。

［対策］

なし。

KFCB31003-E

Setup configurations failed.

[システムの処理]
コマンドは異常終了します。

[対策]
保守員に連絡してください。

KFCB31004-E

aa...aa is not set.

aa...aa：環境変数名

[システムの処理]
プロセスは異常終了します。

[対策]
環境変数 aa...aa に正しい値を設定して，再度実行してください。

KFCB31005-E

Failure to allocate memory.

[システムの処理]
コマンドが終了します。

[対策]
ほかのアプリケーションプログラムを終了させてください。

KFCB31006-E

Fatal error occurred.

[システムの処理]
回復不能エラーが発生しました。プロセスは異常終了します。

[対策]
保守員に連絡してください。

KFCB31007-E

Failure to change path delimiter. path=aa...aa

aa...aa：指定されたパス名

[システムの処理]

プロセスは異常終了します。aa...aa で指定されたパス名のデリミタを変更できませんでした。

[対策]

保守員に連絡してください。

KFCB31008-E

Failure to open definition.

[システムの処理]

コマンドは異常終了します。

[対策]

定義キーが設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合，tsdefvalue コマンドを入力して，システム環境定義を設定してください。

KFCB31009-E

Failure to get definition key.

[システムの処理]

コマンドは異常終了します。

[対策]

定義キーが設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合，tsdefvalue コマンドを入力して，システム環境定義を設定してください。

KFCB31010-E

Failure to get definition.

[システムの処理]

コマンドは異常終了します。

[対策]

定義パラメタが設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合，tsdefvalue コマンドを入力して，システム環境定義を設定してください。

KFCB31011-E

Failure to read definition.

[システムの処理]

コマンドは異常終了します。

[対策]

定義パラメタが設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合，tsdefvalue コマンドを入力して，システム環境定義を設定してください。

KFCB31012-E

KeyName is invalid.

[システムの処理]

コマンドは異常終了します。

[対策]

tslsconf コマンドでシステム環境定義を参照し，正しい定義キー名を指定してください。

KFCB31013-E

tskeycreate failed.

[システムの処理]

コマンドは異常終了します。

[対策]

コマンドに指定した引数を確認して，再度入力してください。

KFCB31014-E

tsdefremove failed.

[システムの処理]

コマンドは異常終了します。

[対策]

コマンドに指定した引数を確認して，再度入力してください。

KFCB31015-E

tskeyremove failed.

[システムの処理]

コマンドは異常終了します。

[対策]

コマンドに指定した引数を確認して，再度入力してください。

KFCB31016-E

Cannot open Key.

[システムの処理]

コマンドは異常終了します。

[対策]

tslsconf コマンドでシステム環境定義を参照し，正しい定義キー名を指定してください。

KFCB31017-E

tsdefvalue failed.

[システムの処理]

コマンドは異常終了します。

[対策]

コマンドに指定した引数を確認して，再度入力してください。

KFCB31018-E

mkdir error. path=aa...aa

aa...aa：ディレクトリ名

[システムの処理]

プロセスは異常終了します。

[対策]

環境変数 TPSPOOL，TPFS に設定したディレクトリが正しいかどうか，および環境変数 TPSPOOL，TPFS に設定したディレクトリを作成するパーミッションがあるかどうかを確認してください。

KFCB31019-E

rmdir error. path=aa...aa

aa...aa：ディレクトリ名

[システムの処理]

プロセスは異常終了します。

[対策]

環境変数 TPSPOOL，TPFS に設定したディレクトリが正しいか，および環境変数 TPSPOOL，TPFS に設定したディレクトリを削除するパーミッションがあるかどうかを確認してください。

KFCB31020-E

File copy error. filename=aa...aa

aa...aa：ファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31021-E

File access error. filename=aa...aa

aa...aa：ファイル名

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31022-E

Parameter is invalid.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31023-I

Starting system setup configurations.

[システムの処理]

処理を続けます。

[対策]

なし。

KFCB31024-E

$TPSPOOL or $TPFS directory already exists.

[システムの処理]

$TPSPOOL，または $TPFS ディレクトリがすでに存在しているため，tssetup コマ

ンドの処理を中断しました。

[対策]

環境変数 TPSPOOL または TPFS で設定されるディレクトリがすでに存在するため，次のどれかの操作をしてください。

1. 環境変数 TPSPOOL または TPFS の値が正しいかどうか確認する。
2. 実行環境を再作成する場合は，-i オプションを指定して tssetup コマンドを実行する。
3. -i オプションを指定した tssetup コマンドでセットアップした環境を作成し直す場合は，環境変数 TPFS に設定されているディレクトリを削除してから再度コマンドを実行する。

KFCB31025-E

Cannot share $TPFS directory.

[システムの処理]

コマンドは異常終了します。

[対策]

環境変数 TPFS で設定されるディレクトリを共用する場合，同じバージョンの TPBroker で共用するように環境変数を設定したあとで，tssetup コマンドを再度実行してください。環境変数 TPFS で設定されるディレクトリを共用する必要がない場合，-i オプションを指定しないで tssetup コマンドを再度実行してください。

KFCB31026-E

$TPB_TRN_TRACE_PATH directory already exists.

[システムの処理]

$TPB_TRN_TRACE_PATH ディレクトリがすでに存在しているため，tssetup コマンドの処理を中断しました。

[対策]

環境変数 TPB_TRN_TRACE_PATH の値が正しいかどうかを確認してください。実行環境を再作成する場合，-d オプションを指定した tssetup コマンドを実行して以前の実行環境を削除してください。

KFCB31033-W

A definition (aa...aa) is irregular.

reason:bb...bb

aa...aa：定義パラメタ

bb...bb：詳細情報

Irregular Defname

TPBroker で規定されていない定義パラメタ名が指定されています。

Invalid Combination

定義パラメタ名と，定義キー名の組み合わせが一致していません。

Invalid Type

定義パラメタ名は TPBroker で規定されていますが，指定値タイプが誤っています。

[システムの処理]

コマンドは，処理を続行します。

[対策]

tsdefremove コマンドで誤ったシステム環境定義を削除して，tsdefvalue コマンドで正しいシステム環境定義を設定してください。

KFCB31200-E

aa...aa function error. rmid=bb...bb rc=cc...cc

aa...aa：XA 関数名

bb...bb：XA 関数を発行したリソースマネージャの実行環境での ID

cc...cc：XA 関数のリターンコード

[システムの処理]

関数およびリターンコードに応じて適切な処理をします。場合によっては，プロセスを異常終了させることがあります。

[対策]

表示された XA 関数名およびリターンコードを基に，接続しているリソースマネージャの設定および環境を見直してください。xa_open や xa_close でのエラーは，リソースマネージャ定義の /OTS/RM/RMn/xa_open_string_info または /OTS/RM/RMn/xa_close_string_info 定義，および環境変数を見直してください。
そのほか，XA 関数のエラーについては，接続するリソースマネージャのマニュアルなどを参照してください。
また，TPBroker やリソースマネージャをバージョンアップした場合，tslnkrm コマンドでリソースマネージャの再登録や，ユーザアプリケーションに再びリンクしていないときにもこのエラーが発生します。TPBroker やリソースマネージャのバージョンアップ後の場合，tslnkrm コマンドの再度実行やユーザアプリケーションに再びリンクしてください。

KFCB31201-E

Cannot add branch to current transaction.

[システムの処理]

リクエストコールの結果として CosTransactions::Unavailable 例外を発生させます。

[対策]

トランザクション定義 OTS/TM/max_crm_branch_count の設定値を大きくしてください。

KFCB31202-E

Cannot initialize the status file.

[システムの処理]

OTS デーモンが終了します。

[対策]

環境変数 TPFS，TPSPOOL やアクセス権限などを見直してください。または，tssetup コマンドを再度入力してください。tssetup コマンドを実行すると，定義内容やリソースマネージャの登録状態などが初期化されるので注意が必要です。

KFCB31203-E

Cannot recover transaction status.

［システムの処理］

OTS デーモンが終了します。

[対策]

トランザクションステータスファイルのオープンまたは読み込みができません。環境変数 TPFS，TPSPOOL やアクセス権限などを見直してください。または，tssetup コマンドを再度入力してください。tssetup コマンドを実行すると，定義内容やリソースマネージャの登録状態などが初期化されるので注意が必要です。

KFCB31204-E

Failed to read XA open/close strings.

[システムの処理]

そのプロセスでのリソースマネージャに対する xa_open または xa_close が行われません。プロセスによっては異常終了することがあります。

[対策]

リソースマネージャ定義の /OTS/RM/RMn/xa_open_string_info または /OTS/RM/RMn/xa_close_string_info の設定値を見直してください。または，環境変数 TPRMINFO の値を見直してください。

KFCB31205-E

Internal error. pid=aa...aa tid=bb...bb info=cc...cc code=dd...dd

aa...aa：プロセス ID

bb...bb：スレッド ID

cc...cc：保守情報 1

dd...dd：保守情報 2

［システムの処理］
プロセスは異常終了します。

［対策］
保守員に連絡してください。

KFCB31206-E

The XA routine return code has an invalid value.

［システムの処理］
プロセスは異常終了します。

［対策］
接続しているリソースマネージャの設定または環境を見直してください。

KFCB31207-E

The xa_ready return code has an invalid value. rmid=aa...aa rc=bb...bb

aa...aa：XA 関数を発行したリソースマネージャの実行環境での ID

bb...bb：XA 関数のリターンコード

［システムの処理］
プロセスは異常終了します。

［対策］
接続しているリソースマネージャの設定または環境を見直してください。

KFCB31208-E

Too many transaction branches. (/OTS/TM/process_count)

［システムの処理］
OTSCurrent::begin()，OTSFactory::create()，またはリクエストコールの結果として IMP_LIMIT 例外を発生させます。

［対策］

トランザクション定義 /OTS/TM/process_count の設定値を大きくしてください。または，リソースマネージャおよびリソースオブジェクトが起動していることを確認してください。

KFCB31209-E

Invalid XA switch.

[システムの処理]
プロセスは異常終了します。

[対策]
新しい XA switch をユーザアプリケーションプログラムにリンクしてください。

KFCB31210-E

xa_ready function error. rmid=aa...aa rc=bb...bb

aa...aa：XA 関数を発行したリソースマネージャの実行環境での ID

bb...bb：XA 関数のリターンコード

[システムの処理]
リターンコードに応じて適切な処理を行います。場合によっては，プロセスを異常終了させることがあります。

[対策]
リターンコードを基に，接続しているリソースマネージャの設定および環境を見直してください。

KFCB31211-E

Fatal error. pid=aa...aa tid=bb...bb info=cc...cc code=dd...dd

aa...aa：プロセス ID

bb...bb：スレッド ID

cc...cc：保守情報 1

dd...dd：保守情報 2

[システムの処理]
プロセスを異常終了します。

[対策]
このメッセージの直前のメッセージを基に対策してください。このメッセージの前にほかのメッセージを出力していない場合は，保守員に連絡してください。

KFCB31212-E

Fatal error. info=aa...aa code=bb...bb

aa...aa：保守情報１

bb...bb：保守情報２

［システムの処理］

プロセスを異常終了します。

［対策］

このメッセージの直前のメッセージを基に対策してください。このメッセージの前にほかのメッセージを出力していない場合は，保守員に連絡してください。

KFCB31213-E

Failed to initialize.

［システムの処理］

プロセスを異常終了します。

［対策］

このメッセージの直前のメッセージを基に対策してください。このメッセージの前にほかのメッセージを出力していない場合は，保守員に連絡してください。

KFCB31214-E

Write to transaction status file failed.

［システムの処理］

プロセスを異常終了します。

［対策］

このメッセージの直前のメッセージを基に対策してください。このメッセージの前にほかのメッセージを出力していない場合は，保守員に連絡してください。

KFCB31215-E

Maximum retry achieved for XA function call.

［システムの処理］

アプリケーションプログラムの場合，プロセスを異常終了します。回復デーモンの場合，一定時間経過後，再度処理を行います。

［対策］

リソースオブジェクトが存在するプロセスが異常終了している場合，そのプロセスを再起動してください。

KFCB31217-E

The xa_complete return code has an invalid value. rmid=aa...aa rc=bb...bb

aa...aa：XA 関数を発行したリソースマネージャの実行環境での ID

bb...bb：XA 関数のリターンコード

［システムの処理］
プロセスは異常終了します。

［対策］
接続しているリソースマネージャの設定または環境を見直してください。

KFCB31218-E

Too many CRM branches. (/OTS/TM/max_crm_branch_count)

［システムの処理］
CosTransactions::Coordinator::register_resource(), CosTransactions::Coordinator::register_subtran_aware()，またはリクエストコールの結果として CORBA::IMP_LIMIT 例外を発生させます。

［対策］
トランザクション定義 /OTS/TM/max_crm_branch_count の設定値を大きくしてください。

KFCB31219-E

Unable to find XA switch list.

［システムの処理］
プロセスは異常終了します。

［対策］
トランザクション制御用オブジェクトをアプリケーションプログラムにリンクしてください。

KFCB31220-W

Creating a new transaction has not been allowed.

［システムの処理］
CORBA::NO_PERMISSION または CORBA::TRANSACTION_ROLLEDBACK 例外を発生させます。

［対策］
OTS が停止しているときに，新しいトランザクションを作成しようとしました。
tsstart コマンドで OTS を再開始してください。

KFCB31221-E

/OTS/RM/aa...aa/set_xa_open_scope is bb...bb.

aa...aa：リソースマネージャ名

bb...bb：設定されている定義値

[システムの処理]

OTS デーモンが終了します。

[対策]

定義値が不正です。正しい値に設定し直してください。

KFCB31222-E

Too many transaction branches. reason=aa...aa

aa...aa：理由コード
/OTS/RCV/set_startup_recovery_skip

[システムの処理]

Current オブジェクトの begin オペレーション，TransactionFactory オブジェクトの create オペレーション，またはリクエストコールの結果として TRANSACTION_ROLLEDBACK 例外を発生させます。

[対策]

トランザクション定義 /OTS/RCV/set_startup_recovery_skip に 1 が指定されていますが，トランザクションブランチが多過ぎるため，OTS 再起動時のトランザクション回復処理が完了していません。時間をおいて再度実行してください。回復できない場合は，リソースマネージャおよびリソースオブジェクトが起動しているかどうかを確認してください。

KFCB31224-W

Transaction was ignored (XA func=aa....aa, rc=bb....bb)

aa...aa：XA 関数名

bb...bb：XA 関数のリターン値

[システムの処理]

XA 関数でエラーが発生したトランザクションブランチをロールバック決着したものとみなします。

[対策]

なし。（エラーが発生した XID を確認したい場合は，$TPSPOOL/log/XATrace を参照して下さい。）

KFCB31226-E

aa...aa function error. xid=bb...bb rmid=cc...cc rc=dd...dd

aa...aa: 発行した XA 関数名

bb...bb: 失敗したトランザクション ID

cc...cc: エラーを返した rmid

dd...dd:RM から戻って来たリターンコード

[意味]

aa...aa 関数でエラーが発生しました。

[システムの処理]

/OTS/RCV/set_recover_retry_count で指定した回数全面回復処理を繰り返します。

[対策]

リソースマネージャが起動されているか又は，デーモンに与える RM のユーザ権限を見直してください。

KFCB31227-E

Transaction recovery processing is not completed. Creating a new transaction has not been allowed.

[意味]

トランザクション回復処理が完了していません。新しいトランザクションを作成することが出来ません。

[システムの処理]

CORBA::NO_PERMISSION または CORBA::TRANSACTION_ROLLEDBACK 例外を発生させます。

[対策]

KFCB31512-I メッセージが発行されてから新しいトランザクションを開始してください。

KFCB31228-I

xid=aa...aa info=bb...bb code1=cc...cc

aa...aa:XID

bb...bb: 保守資料

cc...cc: 保守資料

[意味]

直前のメッセージを出力したトランザクションの XID を出力します。

[システムの処理]
何もしません。

[対策]
なし。

KFCB31233-I

Heuristic decision was ignored. xid=aa...aa info=bb...bb code1=cc...cc code2=dd...dd

aa...aa：ヒューリスティック状態が無視されたトランザクションブランチの XID（トランザクション ID）

bb...bb：保守情報 1

cc...cc：保守情報 2

dd...dd：保守情報 3

[システムの処理]
2 相コミットの 1 相目でヒューリスティックが発生しましたが，無視されました。

[対策]
ご使用のリソースマネージャのマニュアルを参照して，原因を取り除いてください。

KFCB31400-E

/OTS/completion_process_count is aa...aa

aa...aa：現在の定義値

[システムの処理]
OTS デーモンは終了します。

[対策]
定義値が不正です。正しい値に設定し直してください。

KFCB31401-E

/OTS/recovery_process_count is aa...aa

aa...aa：現在の定義値

[システムの処理]
OTS デーモンは終了します。

[対策]
定義値が不正です。正しい値に設定し直してください。

KFCB31402-E

/OTS/RCV/set_retry_time is aa...aa

aa...aa：現在の定義値

［システムの処理］

OTS デーモンは終了します。

［対策］

定義値が不正です。正しい値に設定し直してください。

KFCB31403-E

/OTS/RM/set_xa_async_interval is aa...aa

aa...aa：現在の定義値

［システムの処理］

OTS デーモンは終了します。

［対策］

定義値が不正です。正しい値に設定し直してください。

KFCB31404-E

/OTS/TM/max_crm_branch_count is aa...aa

aa...aa：現在の定義値

［システムの処理］

OTS デーモンは終了します。

［対策］

定義値が不正です。正しい値に設定し直してください。

KFCB31405-E

/OTS/TM/process_count is aa...aa

aa...aa：現在の定義値

［システムの処理］

OTS デーモンは終了します。

［対策］

定義値が不正です。正しい値に設定し直してください。

KFCB31406-E

aa...aa is already registered.

aa...aa：リソースマネージャ名

[システムの処理]

tslnkrm または tsmkobj コマンドは終了します。

[対策]

指定したリソースマネージャ名が TPBroker システムにすでに登録されています。登録されているリソースマネージャ名を tslsrm コマンドで確認し，正しいリソースマネージャ名を指定してください。

KFCB31407-E

OTS definition is illegal.

[システムの処理]

OTS デーモンは終了します。

[対策]

システム環境定義に誤りがありました。定義内容を確認してください。

KFCB31408-E

Cannot access RM information.

[システムの処理]

tslsrm コマンドは終了します。

[対策]

tslnkrm コマンドがリソースマネージャ情報をアップデートしています。tslnkrm コマンドが終了してから tslsrm コマンドを実行してください。

KFCB31409-E

Cannot create RM information.

[システムの処理]

tslnkrm または tsmkobj コマンドは終了します。

[対策]

$TPSPOOL ディレクトリに書き込み権限があるかどうかを確認してください。
また，$TPSPOOL ディレクトリがあるディスクの容量に余裕があるかどうかを確認してください。

KFCB31410-E

Cannot delete aa...aa.

aa...aa：リソースマネージャ名

[システムの処理]

tslnkrm コマンドは終了します。

[対策]

表示された名称のリソースマネージャを削除できません。

KFCB31411-E

Cannot find $TPDIR/bin/TP_CC_DB.

[システムの処理]

tslnkrm または tsmkobj コマンドは終了します。

[対策]

環境変数 TPDIR の値が正しく設定されているかどうかを確認してください。

KFCB31412-E

Cannot open $TPDIR/bin/TP_CC_DB.

[システムの処理]

tslnkrm または tsmkobj コマンドは終了します。

[対策]

$TPDIR/bin/TP_CC_DB ファイルの読み取り権限があるかどうかを確認してください。

KFCB31413-E

Cannot read the information about the compiler.

[システムの処理]

tslnkrm または tsmkobj コマンドは終了します。

[対策]

$TPDIR/bin/TP_CC_DB ファイルが壊れている可能性があります。TPBroker をインストールし直してください。

KFCB31415-E

Recovery definition is illegal.

[システムの処理]

OTS デーモンは終了します。

[対策]

Recovery の定義に誤りがありました。定義内容を確認してください。

KFCB31416-E

RM definition is illegal.

[システムの処理]

OTS デーモンは終了します。

[対策]

リソースマネージャの定義に誤りがありました。定義内容を確認してください。

KFCB31417-E

RM information has changed.

[システムの処理]

リソースマネージャ情報を前回の状態に回復し，OTS デーモンを起動します。

[対策]

前回正常終了していないにもかかわらず，リソースマネージャ情報が変更されています。リソースマネージャ情報を変更したい場合は，強制正常開始モードで起動してください。

KFCB31418-E

System definition has changed.

[システムの処理]

システム環境定義の値が変更されていますが，前回起動時の定義値を有効にして，OTS デーモンを起動します。

[対策]

変更後のシステム環境定義の値を有効にする場合は，次のどちらかの方法で OTS を再起動してください。

1. tsstop コマンドで OTS を正常終了してから，tsstart コマンドで OTS を正常開始する。
2. 任意の方法で OTS を停止してから，-f オプションを指定した tsstart コマンドで OTS を強制正常開始する。

ただし，2. の方法ではトランザクションの情報を引き継げないため，1. の方法で再起動することをお勧めします。

KFCB31419-E

System definition is illegal.

[システムの処理]

OTS デーモンは終了します。

[対策]

システム環境定義に誤りがありました。定義内容を確認してください。

KFCB31420-E

The number of RM's exceeded.

[システムの処理]

OTS デーモンは終了します。

[対策]

TPBroker システムに登録されているリソースマネージャが多過ぎます。tslnkrm コマンドを使用して，使用しないリソースマネージャを TPBroker システムから削除してください。

KFCB31421-E

System definitions not set.

[システムの処理]

コマンドは終了します。

[対策]

環境変数 TPSPOOL の値が正しく設定されているかどうかを確認してください。または，tssetup コマンドを実行して，実行環境を作成してください。

KFCB31422-E

Transaction Service is already online.

[システムの処理]

tsstart コマンドは終了します。

[対策]

TPBroker システムがすでにオンライン状態です。tslnkrm コマンドが実行中かどうかを確認してください。または，$TPSPOOL ディレクトリ下のファイルとディレクトリのアクセス権限があることを確認してください。

KFCB31423-E

Transaction Service is not online.

［システムの処理］

tsstop，tslstrn，tscommit，または tsrollback コマンドが終了します。

［対策］

TPBroker システムが開始されていません。システムを開始したあとにコマンドを実行してください。

KFCB31424-E

Transaction Service is online.

［システムの処理］

tslnkrm コマンドが終了します。

［対策］

TPBroker システムを終了したあとに，tslnkrm コマンドを実行してください。

KFCB31425-E

Transaction definition is illegal.

［システムの処理］

OTS デーモンは終了します。

［対策］

トランザクションの定義に誤りがありました。定義内容を確認してください。

KFCB31426-E

tslnkrm command is now running, Transaction Service cannot start.

［システムの処理］

OTS デーモンは終了します。

［対策］

同一環境で tslnkrm コマンドが実行されています。tslnkrm コマンドが終了したあとに，システムを起動してください。

KFCB31429-E

aa...aa is illegal file.

aa...aa：ファイル名

［システムの処理］

処理を打ち切り，プロセスを異常終了させます。

［対策］

ファイル名に応じて，次に示す内容に従って対策してください。

ファイル名	対策
$TPDIR/bin/TP_CC_DB $TPDIR/bin/TP_CC_DB_FP $TPDIR/bin/TP_CC_DB_FP_R $TPDIR/bin/TP_CC_DB_R	使用している共用ライブラリ，または DLL が $TPDIR ディレクトリ下のものではありません。 ご使用の OS ごとに次の環境変数が $TPDIR/lib に設定されているかどうかを確認してください。 • LD_LIBRARY_PATH（Solaris および Linux） • LIBPATH（AIX） • PATH（Windows） • SHLIB_PATH（HP-UX） 再設定後，コマンドを再度実行してください。
その他	保守員に連絡してください。

KFCB31430-E

Fatal error occurred. (aa...aa,bb...bb)

aa...aa：保守情報 1

bb...bb：保守情報 2

［システムの処理］

プロセスを異常終了させます。

［対策］

保守員に連絡してください。

KFCB31432-E

Bad parameter::aa...aa.

aa...aa：指定されたコマンドオプション

［システムの処理］

プロセスは異常終了します。

［対策］

コマンドに異常なオプションが設定されました。正しいオプションを設定して再度実行してください。

KFCB31433-I

No transactions to process.

［システムの処理］

tslstrn，tscommit，または tsrollback コマンドは終了します。処理対象のトランザクションが見つかりませんでした。

［対策］

なし。

KFCB31434-E

Command error.

［システムの処理］

tscommit，tsrollback，または tslstrn コマンドの実行に失敗しました。

［対策］

OTS デーモンが出力するメッセージ，またはコマンドが出力するほかのメッセージに従って対策してください。

KFCB31435-E

Not enough memory.

［システムの処理］

プロセスは異常終了します。

［対策］

メモリが不足しています。ほかのアプリケーションプログラムを終了させて，再度実行してください。

KFCB31436-E

Unexpected flags. (aa...aa,bb...bb)

aa...aa：保守情報 1

bb...bb：保守情報 2

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB31437-E

Transaction timeout occurred. (GID=aa...aa BID=bb...bb)

aa...aa：トランザクションブランチのグローバル ID

bb...bb：トランザクションブランチのブランチ ID

[システムの処理]

該当トランザクションを処理していたプロセスを強制停止します。設定したトランザクションタイムアウト時間が経過したため，該当トランザクションを処理していたプロセスを強制停止します。該当トランザクションは，自動的にロールバックされます。

[対策]

該当アプリケーションプログラムで設定しているトランザクションタイムアウト値を見直してください。また，タイムアウトが発生した要因を取り除いてください。

KFCB31438-E

Cannot recover transaction branches.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31439-E

An error occurred in tslnkrm command.

[システムの処理]

コマンドは異常終了します。

[対策]

保守員に連絡してください。

KFCB31440-E

Unexpected server status. (aa...aa,bb...bb)

aa...aa：保守情報 1

bb...bb：保守情報 2

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31441-E

$TPSPOOL directory does not exist.

[システムの処理]

OTS デーモン，コマンドが終了します。

[対策]

環境変数 TPSPOOL の値が正しく設定されているかどうかを確認してください。または，tssetup コマンドを実行して，TPBroker の実行環境を作成してください。

KFCB31442-E

Cannot create $TPSPOOL/.command.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31443-E

Cannot create $TPSPOOL/.otsd.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31444-I

Starting OTS Daemon. pid=aa...aa date=bb...bb

aa...aa：OTS デーモンのプロセス ID

bb...bb：OTS デーモンの開始時刻

[システムの処理]

OTS デーモンの開始処理を始めました。

[対策]

なし。

KFCB31446-E

System definition error.

[システムの処理]

OTS デーモンが終了します。

[対策]

システム環境定義に誤りがありました。定義内容を確認してください。

KFCB31447-E

aa...aa is a directory.

aa...aa：ディレクトリ名

[システムの処理]

プロセスは異常終了します。

[対策]

tssetup コマンドをやり直してから，プロセスを再起動してください。

KFCB31448-E

Cannot start monitoring process.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31449-I

Transaction Service has started.

[システムの処理]

トランザクションサービスの開始処理が完了しました。

[対策]

なし。

KFCB31450-E

Unexpected termination. (aa...aa,bb...bb)

aa...aa：保守コード 1

bb...bb：保守コード 2

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31451-I

Transaction Service has ended.

[システムの処理]

トランザクションサービスが終了しました。

[対策]

なし。

KFCB31452-E

Cannot create Transaction Control Tables.

[システムの処理]

OTS デーモンは終了します。

[対策]

トランザクション管理テーブルの作成に失敗しました。トランザクション処理を行うアプリケーションプログラムが起動している場合，一度停止してください。

KFCB31453-E

Cannot start Completion Daemon.

[システムの処理]

プロセスは異常終了します。

[対策]

決着デーモンの起動時にエラーが発生しました。設定が間違っている可能性があります。環境変数の設定およびシステム環境定義を見直してください。

KFCB31454-E

Cannot start Recovery Daemon.

[システムの処理]

プロセスは異常終了します。

[対策]

回復デーモンの起動時にエラーが発生しました。設定が間違っている可能性があります。環境変数の設定およびシステム環境定義を見直してください。

KFCB31455-E

Cannot start Concurrency Control Daemon.

[システムの処理]

プロセスは異常終了します。

[対策]

保守員に連絡してください。

KFCB31456-E

Cannot find a directory, aa...aa.

aa...aa：ディレクトリ名

［システムの処理］

コマンドは異常終了します。

［対策］

tssetup コマンドをやり直してからコマンドを再度実行してください。

KFCB31457-E

Cannot find compiler information.

［システムの処理］

コマンドは異常終了します。

［対策］

tssetup コマンドをやり直してからコマンドを再度実行してください。

KFCB31458-E

Cannot create a new TP_RM_DB file.

［システムの処理］

コマンドは異常終了します。

［対策］

環境変数 TPSPOOL に設定しているディスク領域が不足していないかどうかを確認してください。領域が不足している場合，必要な領域を確保してから再度コマンドを実行してください。領域が十分ある場合，tssetup コマンドをやり直してからプロセスを再起動してください。

KFCB31459-E

aa...aa is not registered.

aa...aa：リソースマネージャ名

［システムの処理］

tslnkrm または tsmkobj コマンドは終了します。

［対策］

指定したリソースマネージャは，現在の実行環境に登録されていません。現在登録されているリソースマネージャ名を tslsrm コマンドで確認してください。また，未登録の場合，tslnkrm コマンドで登録してください。

KFCB31460-E

No RM defined to this system.

［システムの処理］

コマンドは異常終了します。

［対策］

tssetup コマンドをやり直してから再度コマンドを実行してください。

KFCB31462-I

Starting Completion Daemon. pid=aa...aa date=bb...bb

aa...aa：決着デーモンのプロセス ID

bb...bb：決着デーモンの開始時刻

［システムの処理］

決着デーモンの開始処理を始めました。

［対策］

なし。

KFCB31463-I

Completion Daemon has started.

［システムの処理］

決着デーモンの開始処理が終了しました。

［対策］

なし。

KFCB31464-I

Completion Daemon has ended.

［システムの処理］

決着デーモンが終了しました。

［対策］

なし。

KFCB31465-I

Starting Recovery Daemon. pid=aa...aa date=bb...bb

aa...aa：回復デーモンのプロセス ID

bb...bb：回復デーモンの開始時刻

［システムの処理］

回復デーモンの開始処理を始めました。

［対策］

なし。

KFCB31466-I

Recovery Daemon has started.

［システムの処理］

回復デーモンの開始処理が終了しました。

［対策］

なし。

KFCB31467-I

Recovery Daemon has ended.

［システムの処理］

回復デーモンが終了しました。

［対策］

なし。

KFCB31474-E

Cannot find $TPDIR/bin/TP_CC_DB_R.

［システムの処理］

コマンドは異常終了します。

［対策］

$TPDIR/bin/TP_CC_DB_R ファイルが存在するかどうかを確認してください。存在しない場合，再インストールしてください。

KFCB31475-E

Cannot open $TPDIR/bin/TP_CC_DB_R.

［システムの処理］

コマンドは異常終了します。

［対策］

$TPDIR/bin/TP_CC_DB_R ファイルに読み取り権限があるかどうかを確認してください。

KFCB31476-E

Cannot find aa...aa compiler information in TP_CC_DB file.

aa...aa：保守情報

［システムの処理］

コマンドは異常終了します。

［対策］

tssetup コマンドをやり直してから再度コマンドを実行してください。

KFCB31477-E

Cannot move files to aa...aa.

aa...aa：ファイル名

［システムの処理］

コマンドは異常終了します。

［対策］

ファイル aa...aa に書き込み権限があるかどうかを確認してください。または，環境変数 TPSPOOL に設定しているディスク領域が不足していないかどうかを確認してください。領域が不足している場合，必要な領域を確保してから再度実行してください。

KFCB31478-E

Cannot add/delete RMs without -f option.

［システムの処理］

tslnkrm コマンドは終了します。

［対策］

TPBroker が前回正常に終了していません。TPBroker を再開始して，未決着のトランザクションがあれば決着をしてから tsstop コマンドで正常終了し，そのあとに tslnkrm コマンドを実行してください。または，tslnkrm コマンドに -f オプションを指定して実行してください。この場合，未決着のトランザクションがあっても次回の起動時にそのトランザクションは回復されません。

KFCB31479-E

Cannot create daemon aa...aa.

aa...aa：作成に失敗したデーモンの名称

［システムの処理］

tslnkrm コマンドが終了します。

[対策]

デーモンの作成に失敗しました。tslnkrm コマンドのオプションが正しいかどうかを確認してください。環境変数 PATH（Windows 版の場合）が %TPDIR%¥bin に正しく設定されているかどうかを確認してください。または，環境変数 SHLIB_PATH（HP-UX 版の場合），環境変数 LD_LIBRARY_PATH（Solaris および Linux 版の場合），環境変数 LIBPATH（AIX 版の場合）が $TPDIR/lib に正しく設定されているかどうかを確認してください。

KFCB31480-E

A compiler error has occurred. (aa...aa)

aa...aa：コンパイラの名称

[システムの処理]

tslnkrm または tsmkobj コマンドは終了します。

[対策]

コンパイラのエラーメッセージに従って対処してください。
Windows 版の場合，環境変数 INCLUDE が正しく設定されているかどうかを確認してください。

KFCB31481-E

Timeout has occurred.

[システムの処理]

tsstart または tsstop コマンドは終了します。TPBroker の開始処理，または終了処理に時間が掛かったために，コマンドの処理を中断しました。TPBroker は開始処理，または終了処理を続けています。

[対策]

処理が終了するまでお待ちください。

KFCB31482-E

aa...aa terminated.

aa...aa：異常終了したデーモンの名称

[システムの処理]

TPBroker の開始処理を中断し，終了します。TPBroker の開始処理中にデーモンが異常終了しました。

[対策]

データベースなどと XA 連携している場合，設定に誤りがある可能性があります。

環境変数やシステム環境定義などを見直してください。

KFCB31483-E

Process terminated. (pid=aa...aa)

aa...aa：異常終了したプロセスのプロセス ID

［システムの処理］

プロセス ID が aa...aa のプロセスが異常終了したため，トランザクションの回復処理をします。異常終了したプロセスが処理していたトランザクションは，自動的に回復されます。

［対策］

異常終了したプロセス上にリソースオブジェクトを作成している場合は，必要に応じてプロセスを再起動してください。
また，プロセスが正常停止した場合でも，そのプロセスが処理していたトランザクションが決着されていなかったときは，このメッセージが出力されます。アプリケーションプログラムのトランザクションの開始と決着の対応を見直してください。

KFCB31484-I

Transaction recovered. (GID=aa...aa BID=bb...bb)

aa...aa：回復したトランザクションブランチのグローバル ID

bb...bb：回復したトランザクションブランチのブランチ ID

［システムの処理］

表示されたトランザクションブランチを回復しました。

［対策］

なし。

KFCB31485-I

Transaction cannot be recovered. (GID=aa...aa BID=bb...bb)

aa...aa：回復できなかったトランザクションブランチのグローバル ID

bb...bb：回復できなかったトランザクションブランチのブランチ ID

［システムの処理］

表示されたトランザクションブランチを回復できませんでした。回復できなかったブランチは，ほかのブランチの決着処理時に決着されるか，あとで再度回復処理をして決着されます。

［対策］

TPBroker が決着できないトランザクションブランチは，tscommit または

tsrollback コマンドを使用して決着方法を指示してください。

KFCB31486-I

tsstart successful.

[システムの処理]
TPBroker の開始処理が正常に終了しました。

[対策]
なし。

KFCB31487-I

tsstop successful.

[システムの処理]
TPBroker の終了処理が正常に終了しました。

[対策]
なし。

KFCB31488-W

tsstop has forced the termination of Transaction Service.

[システムの処理]
TPBroker を強制的に終了しました。TPBroker を指定したモードで終了できなかったため，強制的に停止しました。

[対策]
なし。

KFCB31489-I

OTS Daemon already exists.

[システムの処理]
OTS デーモンは終了します。

[対策]
トランザクションサービスがすでに開始しているか，または現在開始中です。新たに開始したい場合，異なる実行環境を使用するか，または tsstop コマンドでオンライン状態のトランザクションサービスを停止してください。

KFCB31490-I

Ending Transaction Service.

[システムの処理]

TPBroker の終了処理を開始します。

[対策]

なし。

KFCB31491-E

Environment is illegal. reason=aa...aa

aa...aa：理由コード

[システムの処理]

プロセスは異常終了します。

[対策]

理由コードに応じて，対策してください。

理由コード	対策
License Information	環境変数 TPDIR，PATH，SHLIB_PATH（HP-UX の場合），LD_LIBRARY_PATH（Solaris および Linux の場合），または LIBPATH（AIX の場合）の設定が正しいか確認してください。
その他	保守員に連絡してください。

KFCB31492-E

The port number of Completion Daemon is already in use. pid=aa...aa port=bb...bb info1=cc...cc

aa...aa：決着デーモンのプロセス ID

bb...bb：決着デーモンが使用するポート番号

cc...cc：保守情報

[システムの処理]

OTS 開始処理を中断し，OTS デーモンは終了します。

[対策]

決着デーモンで使用するポート番号が，すでにほかのプロセスによって使用されているか，OS で使用できないと判断された状態です。netstat コマンドや services ファイルで該当するプロセスが使用するポート番号を調べ，使用されていないポート番号に変更してください。または，トランザクション定義 /OTS/completion_process_port_base の値に，未使用のポート番号を設定してください。

KFCB31493-W

Definition read error. (aa...aa) The default value (bb...bb) is assumed.

aa...aa：定義パラメタ

bb...bb：仮定されるデフォルト値

［システムの処理］

デフォルト値 bb...bb が指定されたものと仮定して処理を続行します。

［対策］

定義パラメタ aa...aa が設定されているかどうかを tslsconf コマンドで確認してください。設定されていない場合，tsdefvalue コマンドを使用して定義パラメタ aa...aa を設定してください。

KFCB31494-W

A definition (aa...aa=bb...bb) is invalid. cc...cc is assumed.

aa...aa：定義パラメタ

bb...bb：定義に設定された値

cc...cc：定義値として仮定される値

［システムの処理］

値として cc...cc が設定されたものと仮定して処理を続行します。

［対策］

定義パラメタ aa...aa に無効な設定がされています。tslsconf コマンドを実行して設定内容を見直し，tsdefvalue コマンドを実行して正しい値を設定してください。

KFCB31495-E

Deadlock detected; OTS Daemon aborted.

［システムの処理］

OTS デーモンは異常終了します。

［対策］

トランザクショナルアプリケーションが，クリティカルな領域をロックしたまま異常終了，またはトランザクションタイムアウトによって強制終了しました。発生原因を取り除き，OTS およびアプリケーションプログラムを再開始してください。

KFCB31496-I

Transaction Service is online.

［システムの処理］

何もしません。トランザクションサービスは現在オンラインです。

[対策]

なし。

KFCB31497-I

Transaction Service is offline.

[システムの処理]

何もしません。トランザクションサービスは現在オフラインです。

[対策]

なし。

KFCB31498-W

aa...aa is not registered.

aa...aa：リソースマネージャ名称

[システムの処理]

指定された aa...aa は，現在の実行環境では登録されていません。コマンドは処理を続けます。

[対策]

なし。

KFCB31499-E

Cannot access shared memory.

[システムの処理]

プロセスは異常終了します。

[対策]

OTS デーモンが開始されていることを確認し，コマンドを再度入力してください。

KFCB31500-E

Cannot start Trace Daemon.

[システムの処理]

トレースデーモンを起動できませんでした。

[対策]

このメッセージの前にほかのメッセージを出力している場合は，そのメッセージを基に対策してください。

このメッセージの前にほかのメッセージを出力していない場合は，保守員に連絡し

てください。

KFCB31501-I

Starting Trace Daemon. pid=aa...aa date=bb...bb

aa...aa：プロセス ID

bb...bb：トレースデーモンの開始時刻

[システムの処理]

トレースデーモンの開始処理を始めました。

[対策]

なし。

KFCB31502-I

Trace Daemon has started.

[システムの処理]

トレースデーモンの開始処理が終了しました。

[対策]

なし。

KFCB31503-I

Trace Daemon has ended.

[システムの処理]

トレースデーモンが終了しました。

[対策]

なし。

KFCB31504-E

Trace definition is illegal.

[システムの処理]

トレースのシステム環境定義に不正があります。トレースデーモンはデフォルトの定義で起動します。

[対策]

なし。

KFCB31505-W

aa...aa has failed. reason=bb...bb info1=cc...cc info2=dd...dd info3=ee...ee pid=ff...ff

aa...aa：デーモン名称

Completion Daemon：決着デーモン

OTS Daemon：OTS デーモン

Recovery Daemon：回復デーモン

Trace Daemon：トレースデーモン

bb...bb：障害理由

CORBA::COMM_FAILURE：CORBA::COMM_FAILURE 例外が発生しました。トランザクション定義 /OTS/set_ipaddr_info または /OTS/completion_process_ipaddr_info に指定された IP アドレスが使用可能かどうかを確認してください。また，トランザクション定義 /OTS/completion_process_port_base に指定されたポート番号ベースから，/OTS/completion_process_count に指定されたプロセス数分のポート番号が使用可能かどうかを確認してください。

cc...cc：保守情報 1

dd...dd：保守情報 2

ポート番号（aa...aa が Completion Daemon で，bb...bb が CORBA::COMM_FAILURE の場合）

ee...ee：保守情報 3

ホスト名または IP アドレス（bb...bb が CORBA::COMM_FAILURE で，ホスト名または IP アドレスが指定されている場合）

ff...ff：プロセス ID

［システムの処理］

デーモンプロセスは異常終了します。OTS サービスも停止します。

［対策］

障害理由を参照して，原因を取り除いてください。

KFCB31506-E

aa...aa downed. pid=bb...bb

aa...aa：デーモン名称

Completion Daemon：決着デーモン

OTS Daemon：OTS デーモン

Recovery Daemon：回復デーモン

Trace Daemon：トレースデーモン

bb...bb：プロセス ID

[システムの処理]

デーモンプロセスは異常終了します。

[対策]

このメッセージの直前のメッセージを基に対策してください。このメッセージの前にほかのメッセージを出力していない場合は，保守員に連絡してください。

KFCB31507-I

$TPFS directory was created.

[システムの処理]

$TPFS ディレクトリがないため，OTS デーモンは $TPFS ディレクトリを作成し，処理を続行します。

[対策]

-i オプションを指定して tssetup コマンドを実行した場合の初回起動時は対策の必要はありません。それ以外の場合，$TPFS ディレクトリがマウントされているか，または $TPFS ディレクトリが削除されていないかを見直してください。

KFCB31509-I

OTS Daemon is terminating process(OTS Application,Daemon of OTS function).

[意味]

OTS デーモンがプロセスを停止しています。

[システムの処理]

OTS デーモンは OTS アプリケーションプログラム及び決着デーモン，回復デーモンを停止します。

[対策]

なし。

KFCB31510-I

OTS Application and Daemon of OTS function was terminated by OTS Daemon.

[意味]

OTS アプリケーションプログラム，および OTS 機能のデーモンが OTS デーモンによってプロセスを停止されました。

[システムの処理]

なし。

[対策]

なし。

KFCB31511-E

Maximum retry achieved for XA function call. (/OTS/RCV/set_recover_retry_count)

［意味］

システム定義 /OTS/RCV/set_recover_retry_count に指定されている最大のリトライ回数 XA 関数の発行を行いました。

［システムの処理］

OTS の起動に失敗します。

［対策］

リソースマネージャが起動されているか確認してください。

KFCB31512-I

xa_recover() function finished. A new transaction can be started.

［意味］

xa_recover() 関数が終了しました。新しいトランザクションを開始できます。

［システムの処理］

回復処理は終了しました。新しいトランザクションを開始できます。

［対策］

なし。

KFCB31900-W

Tracing service cannot be started. pid=aa...aa reason=bb...bb

aa...aa：プロセス ID

bb...bb：理由コード

101

予期しないエラーが発生しました。

102

トレースサービスで使用する共用メモリのアドレスを取得できません。

103

メモリ不足が発生しました。

104

msgget システムコールで，新しいメッセージ待ち行列 ID の割り当てに失敗しました。ipcs コマンドを実行して使用状況を確認してください。不要なメッセージキューがある場合，ipcrm コマンドを実行して削除してください。

105

システムコールでエラーが発生しました。

106

トレースサービスで使用する共用メモリでメモリ不足が発生しました。

112

指定されたトレースサービスは，すでに開始されています。または，トレースサービスは終了していますが，トレースを取得していたプロセスが起動中のため，新たにトレースサービスを開始できません。

114

プロセス内のスレッド生成システムコールでエラーが発生しました。

115

ファイルのオープンでエラーが発生しました。

116

ファイルの検定時，または新規作成時に I/O エラーが発生しました。

[システムの処理]

該当する OTS システムのトレースサービスを中止します。

[対策]

理由コードを参照して，障害の原因を取り除いてください。

KFCB31901-W

Failed to initialize tracing service. pid=aa...aa reason=bb...bb

aa...aa：プロセス ID

bb...bb：理由コード

101：予期しないエラーが発生しました。

102：トレースサービスで使用する共用メモリのアドレスを取得できません。

103：メモリ不足が発生しました。

105：システムコールでエラーが発生しました。

[システムの処理]

該当するプロセスのトレース取得を中止します。

[対策]

理由コードを参照して，障害の原因を取り除いてください。

KFCB31902-W

Failed to output shared memory information. reason=aa...aa

aa...aa：理由コード

101

トレースサービスで使用する共用メモリをダンプしようとしたが，共用メモリが削除された可能性があります。前回の OTS 異常終了後に，OS が再起動され

ていないか，または ipcrm コマンドで共用メモリを削除していないかどうかを確認してください。削除していない場合，システムコールでエラーが発生しました。

117

トレースサービスで使用する共用メモリのダンプファイル出力処理中にメモリ不足が発生しました。

118

トレースサービスで使用する共用メモリのダンプファイル出力処理中のロック処理でエラーが発生しました。

119

トレースサービスで使用する共用メモリのダンプファイル出力処理のファイルアクセスでエラーが発生しました。

120

トレースサービスで使用する共用メモリが壊れています。

[システムの処理]

該当する OTS システムでのトレースサービスは正常に開始されます。ただし，OTS 異常終了時の一部のトレース情報が取得できなかった可能性があります。

[対策]

理由コードを参照して，障害の原因を取り除いてください。

KFCB31903-E

Not enough memory.

[システムの処理]

プロセスは異常終了します。

[対策]

ほかのアプリケーションプログラムを終了してください。

KFCB31904-E

Command argument is invalid.

[システムの処理]

コマンドは異常終了します。

[対策]

コマンド引数が不正です。このメッセージの直後に出力される使用方法のメッセージに従って，再度入力してください。

KFCB31906-E

Command error. (aa...aa,bb...bb)

aa...aa：保守情報 1

bb...bb：保守情報 2

［システムの処理］

プロセスは異常終了します。

［対策］

保守員に連絡してください。

KFCB31907-E

Cannot open tracefile. reason=aa...aa

aa...aa：理由コード

101：予期しないエラーが発生しました。

103：メモリ不足が発生しました。

115：ファイルのオープンに失敗しました。

121：指定したトレースファイル名に誤りがあります。

［システムの処理］

プロセスは異常終了します。

［対策］

理由コードを参照して，障害の原因を取り除いてください。

KFCB31908-W

Failed to read record of the Trace File. reason=aa...aa

aa...aa：理由コード

101：予期しないエラーが発生しました。

103：メモリ不足が発生しました。

116：トレースファイルの I/O エラーが発生しました。

121：指定したトレースファイル名に誤りがあります。

122：レコードの読み込みに失敗しました。

［システムの処理］

レコードを正常に読み込めませんでした。

［対策］

理由コードを参照して，障害の原因を取り除いてください。

KFCB32000-E

Fatal lock error occurred. info=aa...aa

aa...aa：保守情報

［システムの処理］

プロセスを異常終了します。処理中のトランザクションがある場合，決着処理をします。

［対策］

OS のロックに関する何らかの問題が発生しています。その原因を取り除いてください。

KFCB32001-E

Fatal error occurred. The process aborts. pid=aa...aa tid=bb...bb info=cc...cc code1=dd...dd code2=ee...ee

aa...aa：プロセス ID

bb...bb：スレッド ID

cc...cc：保守情報 1

dd...dd：保守情報 2

ee...ee：保守情報 3

［システムの処理］

処理中のトランザクションがある場合，決着処理をします。

［対策］

このメッセージの直前のメッセージを基に対策してください。このメッセージの前にほかのメッセージを出力していない場合は，保守員に連絡してください。

KFCB32002-W

CORBA exception caught. excep=aa...aa pid=bb...bb tid=cc...cc info=dd...dd code1=ee...ee code2=ff...ff

aa...aa：catch した CORBA 例外

CORBA 例外については，マニュアル「TPBroker　プログラマーズガイド」およびマニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

bb...bb：プロセス ID

cc...cc：スレッド ID

dd...dd：保守情報 1

ee...ee：保守情報 2

ff...ff：保守情報 3

[システムの処理]

処理を継続できる場合，継続します。継続できない場合，発生例外や発生場所によって適切な処理をします。

[対策]

なし。

KFCB32003-E

Unexpected CORBA exception caught. excep=aa...aa pid=bb...bb tid=cc...cc info=dd...dd code1=ee...ee code2=ff...ff

aa...aa：catch した CORBA 例外

CORBA 例外については，マニュアル「TPBroker　プログラマーズガイド」およびマニュアル「Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド」を参照してください。

bb...bb：プロセス ID

cc...cc：スレッド ID

dd...dd：保守情報 1

ee...ee：保守情報 2

ff...ff：保守情報 3

[システムの処理]

処理を継続できる場合，継続します。継続できない場合，発生例外や発生場所によって適切な処理をします。

[対策]

なし。

KFCB32004-E

Unknown exception caught. pid=aa...aa tid=bb...bb info=cc...cc code1=dd...dd code2=ee...ee

aa...aa：プロセス ID

bb...bb：スレッド ID

cc...cc：保守情報 1

dd...dd：保守情報 2

ee...ee：保守情報 3

[システムの処理]

処理を継続できる場合，継続します。継続できない場合，プロセスを異常終了します（KFCB32001-E のメッセージが出力されます）。

［対策］

プロセスを異常終了する場合は syslog またはイベントログ，および $TPSPOOL ディレクトリ下のファイルを保存して保守員に連絡してください。

KFCB32005-W

Failed to delete objects. info1=aa...aa info2=bb...bb

aa...aa：保守情報 1

bb...bb：保守情報 2

［システムの処理］

オブジェクトの削除に失敗したが，処理を続けます。

［対策］

メモリリークが生じている可能性があるため，頻発する場合は保守員に連絡してください。

KFCB32006-E

Internal error occurred. reason=aa...aa info=bb...bb code=cc...cc

aa...aa：保守情報 1

bb...bb：保守情報 2

cc...cc：保守情報 3

［システムの処理］

プロセスは異常終了します。

［対策］

syslog またはイベントログ，および $TPSPOOL ディレクトリ下のファイルを保存して保守員に連絡してください。

KFCB32007-W

Transaction created outside the Transaction Service and imported is ignored at the end of transaction branch. info=aa...aa code1=bb...bb code2=cc...cc

aa...aa：保守情報 1

bb...bb：保守情報 2

cc...cc：保守情報 3

［システムの処理］

処理を続けます。

［対策］

なし。

KFCB32008-E

aa...aa is not supported by Transaction Service. info=bb...bb code=cc...cc

aa...aa：サポートされていない機能

TII：TII（Time Independent Invocation）はサポートされていません。

oneway：トランザクショナルな一方送信（一方受信）呼び出しはサポートされていません。

bb...bb：保守情報 1

cc...cc：保守情報 2

［システムの処理］

CORBA::IMP_LIMIT 例外を発生させます。

［対策］

該当する機能は使用しないでください。

KFCB32009-W

Reply check is unavailable. reason=aa...aa

aa...aa：使用できない理由

［システムの処理］

処理を続けます。ただし，応答チェックは行いません。

［対策］

なし。

KFCB32010-E

Failed to allocate memory. info=aa...aa code1=bb...bb code2=cc...cc

aa...aa：保守情報 1

bb...bb：保守情報 2

cc...cc：保守情報 3

［システムの処理］

CORBA::NO_MEMORY 例外を発生させるか，または該当プロセスを異常終了します。

［対策］

使用できる仮想メモリを増やすか，または不要なメモリ確保をしていないかどうかを見直してください。

KFCB32011-E

An exception occurred at the Resource operation. operation=aa...aa excep=bb...bb pid=cc...cc

aa...aa：例外の発生したオペレーション名

bb...bb：発生した例外名

cc...cc：例外が発生したプロセスのプロセス ID

［システムの処理］

トランザクションの決着については，OTS が適切に判断して処理を行います。ただし，commit オペレーションの例外ではリソースオブジェクトに対してオペレーションのリトライを行います。

［対策］

ヒューリスティック例外が発生している場合は，管理者に連絡し，リソースの不整合が生じていないかを確認してください。Resource オブジェクトが存在するプロセスが異常終了している場合，必要に応じてプロセスを再開始してください。

KFCB32012-E

An exception occurred at the Synchronization operation. operation=aa...aa excep=bb...bb pid=cc...cc

aa...aa：例外の発生したオペレーション名

bb...bb：発生した例外名

cc...cc：例外が発生したプロセスのプロセス ID

［システムの処理］

before_completion オペレーションで例外が発生した場合，処理中のトランザクションをロールバックさせます。after_completion オペレーションでの例外は，無視されます。

［対策］

Synchronization オブジェクトが存在するプロセスが異常終了している場合，必要に応じてプロセスを再起動してください。

KFCB32013-E

An exception occurred at the SubtransactionAwareResource operation. operation=aa...aa excep=bb...bb pid=cc...cc

aa...aa：例外の発生したオペレーション名

bb...bb：発生した例外名

cc...cc：例外が発生したプロセスのプロセス ID

[システムの処理]

処理中のサブトランザクションを含むトランザクションファミリーをロールバックします。

[対策]

SubtransactionAwareResource オブジェクトが存在するプロセスが異常終了している場合，必要に応じてプロセスを再起動してください。

KFCB32014-E

Unable to create subtransaction. Number of the subtransaction has reached to maximum in this Coordinator. pid=aa...aa

aa...aa：例外が発生したプロセスのプロセス ID

[システムの処理]

Coordinator::create_subtransaction または Current::begin オペレーションを発行したアプリケーションプログラムに対して CORBA::IMP_LIMIT 標準例外を発生させます。

[対策]

アプリケーションプログラムの構成を見直し，一つの Coordinator に対して生成するサブトランザクションの数を削減してください。上限値は 255 です。

KFCB32015-E

Transaction Service is unavailable. info=aa...aa code1=bb...bb code2=cc...cc

aa...aa：保守情報 1

bb...bb：保守情報 2

cc...cc：保守情報 3

[システムの処理]

プロセスは異常終了します。

[対策]

トランザクションサービスが開始されていないか，またはメモリが不足している可能性があります。トランザクションサービスが開始されていない場合，tsstart コマンドを実行してください。

KFCB32016-E

Cannot start transaction branch. info=aa...aa code1=bb...bb code2=cc...cc

aa...aa：保守情報 1

bb...bb：保守情報 2

cc...cc：保守情報 3

［システムの処理］

プロセスは異常終了します。トランザクションブランチの開始に失敗しました。

［対策］

リソースマネージャの準備が完了していないか，またはトランザクションブランチの数がシステムの上限値を超えた可能性があります。KFCB31208-E が直前に出力されている場合，KFCB31208-E の対策に従ってください。または，リソースマネージャの環境および起動を確認してください。

KFCB32017-I

Heuristic decision was ignored. otid=aa...aa info=bb...bb code1=cc...cc code2=dd...dd

aa...aa：ヒューリスティック状態が無視されたトランザクションブランチの otid（トランザクション ID）

bb...bb：保守情報 1

cc...cc：保守情報 2

dd...dd：保守情報 3

［システムの処理］

次の場合，発生したヒューリスティックが無視されました。

- report_heuristics 引数に false を指定した commit() オペレーション実行
- rollback() オペレーション実行
- commit() オペレーションの延長の 2 相コミットの 1 相目実行

［対策］

ご使用のリソースマネージャのマニュアルを参照して，原因を取り除いてください。

KFCB32018-E

Root transaction associated with Transaction Current object has not completed. The transaction is rolled back. transaction name=aa...aa

aa...aa：ロールバックされたトランザクション名

［システムの処理］

ロールバック後，処理を続けます。

［対策］

TransactionCurrent オブジェクトに結び付いたトランザクションを決着するか，またはサスペンドしてからサーバオブジェクトのオペレーションを終了するように，

アプリケーションプログラムを修正してください。

KFCB32019-E

Cannot preserve transaction information. otid=aa...aa info=bb...bb code1=cc...cc code2=dd...dd

aa...aa：情報の保存に失敗したトランザクションブランチの otid

bb...bb：保守情報 1

cc...cc：保守情報 2

dd...dd：保守情報 3

［システムの処理］

プロセスは異常終了します。トランザクション情報の保存に失敗しました。

［対策］

トランザクションブランチに登録された Synchronization オブジェクト，および SubtransactionAwareResource オブジェクトの数がシステムの許容量を超えました。これらの数を減らすか，インタフェース名，オブジェクト名を短いものに変更してください。

KFCB32020-E

A policy value is invalid. reason=aa...aa value=bb...bb

aa...aa：不正なポリシー値が設定されたポリシータイプ

OTSPolicy：OTSPolicy の値が不正です。

InvocationPolicy：InvocationPolicy の値が不正です。

NonTxTargetPolicy：NonTxTargetPolicy の値が不正です。

bb...bb：UNKNOWN（文字列）

［システムの処理］

オペレーションを停止します

［対策］

Policy の値を確認してください。

KFCB32021-E

A policy type is invalid. reason=aa...aa value=bb...bb

aa...aa：Unknown Policy Type

bb...bb：設定されたポリシー値（整数値）

［システムの処理］

オペレーションを停止します。

[対策]

Policy の値を確認してください。

KFCB32022-E

The combination of OTSPolicy value and InvocationPolicy value is invalid. value1=aa...aa value2=bb...bb

aa...aa：設定された OTSPolicy の値

bb...bb：設定された InvocationPolicy の値

[システムの処理]

オペレーションを停止します。

[対策]

OTSPolicy の値と InvocationPolicy の値の組み合わせが正しいかどうかを確認してください。

KFCB32023-E

OTSPolicy value or InvocationPolicy value is duplicate.

[システムの処理]

オペレーションを停止します。

[対策]

OTSPolicy，または InvocationPolicy を重複して設定していないかどうかを確認してください。

KFCB32024-E

The way to call server don't follow the InvocationPolicy value. Server's InvocationPolicy value is aa...aa.

aa...aa：InvocationPolicy の値

[システムの処理]

オペレーションを停止します。

[対策]

InvocationPolicy の値を確認してください。

KFCB32025-E

Server requires the transactional call. Server's OTSPolicy value is REQUIRES. code=aa...aa

aa...aa：メソッド名

［システムの処理］

オペレーションを停止します。

［対策］

トランザクショナルな呼び出しをしているかどうかを確認してください。

KFCB32026-E

Server requires the non-transactional call. Server's OTSPolicy value is FORBIDS. code=aa...aa

aa...aa：メソッド名

［システムの処理］

オペレーションを停止します。

［対策］

トランザクショナルでない呼び出しをしているかどうかを確認してください。

KFCB32027-E

The TaggedComponent of Server's IOR is invalid. reason=aa...aa

aa...aa：TaggedComponentから読み出した値が不正なポリシータイプ

OTSPolicy：TaggedComponentから読み出したOTSPolicyの値が不正です。

InvocationPolicy：TaggedComponentから読み出したInvocationPolicyの値が不正です。

［システムの処理］

オペレーションを停止します。

［対策］

サーバの処理を見直してください。

KFCB32028-E

Marshaling OTS ServiceContext was failed. code= aa...aa

aa...aa：メソッド名

［システムの処理］

オペレーションを停止します。

［対策］

保守員に連絡してください。

KFCB32200-I

aa...aa

aa...aa：Java プログラムから出力される UAP トレースメッセージ，またはシステムトレースメッセージ

［システムの処理］

なし。

［対策］

Java プログラムから出力される UAP トレースメッセージを確認してください。エラーが発生している場合，Java プログラムを見直してください。TPBroker のエラーメッセージが出力されている場合，メッセージを保存して，保守員へ連絡してください。

KFCB32201-I

aa...aa has started. pid=bb...bb

aa...aa：デーモン名

TransactionContextServer：トランザクションコンテキストサーバ

bb...bb：プロセス ID

［システムの処理］

デーモンの開始処理は完了しました。

［対策］

なし。

KFCB32202-I

aa...aa has stopped. pid=bb...bb

aa...aa：デーモン名

TransactionContextServer：トランザクションコンテキストサーバ

bb...bb：プロセス ID

［システムの処理］

デーモンは終了しました。

［対策］

なし。

KFCB32203-E

aa...aa has failed. reason=bb...bb code=cc...cc pid=dd...dd

aa...aa：デーモン名，またはコマンド名

TransactionContextServer：トランザクションコンテキストサーバ

tsstoptrnctxsv：tsstoptrnctxsv コマンド

bb...bb：次に示す障害理由

NO MEMORY

メモリ不足

Already Started

トランザクションコンテキストサーバはすでに開始しています。複数のトランザクションコンテキストサーバは起動できません。トランザクションコンテキストサーバが存在していることを確認してください。

Not Exist

tsstoptrnctxsv コマンドがトランザクションコンテキストサーバを発見できませんでした。トランザクションコンテキストサーバが起動しているかどうかを確認してください。

Definition

トランザクションコンテキストサーバに関連するシステム環境定義にエラーがあります。システム環境定義を見直してください。

cc...cc：保守情報

dd...dd：プロセス ID

［システムの処理］

デーモン，またはコマンドを異常終了します。

［対策］

障害理由を参照して，原因を取り除いてください。

KFCB32204-E

There is an invalid configuration. name=aa...aa value=bb...bb

aa...aa：不正な定義値が設定されている定義名

bb...bb：不正な定義値

［システムの処理］

該当する定義を無視して，デフォルト値を使用します。

［対策］

該当する定義を確認し，正しい値を設定してください。

KFCB32205-E

Transaction timeout occurred. result=aa...aa transaction_name=bb...bb

aa...aa：ロールバックした結果

RolledBack：トランザクションはロールバックしました。

HeuristicMixed：HeuristicMixed 例外が発生しました。

HeuristicHazard：HeuristicHazard 例外が発生しました。

bb...bb：Coordinator::get_transaction_name() が返したトランザクション名

［システムの処理］

トランザクションコンテキストサーバはトランザクションのタイムアウトを検知し，そのトランザクションをロールバックします。

［対策］

ロールバックの結果が Heuristic 例外である場合，トランザクション状態を確認してください。

KFCB32206-E

Error reason=xx...xx code=yy...yy

Java OTS が出力するメッセージです。詳細は，「12.　Java OTS が出力するメッセージ」を参照してください。

KFCB32207-W

Invalid property value property_key=aa...aa invalid=bb...bb action=cc...cc

aa...aa：プロパティ名

bb...bb：誤ったプロパティ値

cc...cc：アクション

used_default：デフォルト値を使用

ignored：無視

［内容］

プロパティ値を無視しました。プロパティ値が誤っているか，または利用できない状態です。

［システムの処置］

システムはアクションに従って処理を続行します。

［対策］

プロパティ値を確認してください。

KFCB32208-E

aa...aa has failed. reason=bb...bb info1=cc...cc info2=dd...dd info3=ee...ee pid=ff...ff

aa...aa：デーモンまたはコマンド名称

TransactionContextServer：トランザクションコンテキストサーバ

bb...bb：障害理由

CORBA::COMM_FAILURE：CORBA::COMM_FAILURE 例外が発生しました。指定された IP アドレスが使用できない可能性があります。トランザクションコンテキ

ストサーバに関連するシステム環境定義，またはコマンドのオプションを見直してください。

cc...cc：保守情報 1

dd...dd：保守情報 2

ホスト名または IP アドレス（bb...bb が CORBA::COMM_FAILURE で，ホスト名または IP アドレスが指定されている場合）

ee...ee：保守情報 3

ff...ff：プロセス ID

［システムの処理］

デーモンプロセスまたはコマンドは異常終了します。

［対策］

障害理由を参照して，原因を取り除いてください。

KFCB32400-E

The transaction was decided to be heuristic due to an XAResource error.method=aa...aa rc=bb...bb xid=cc...cc info=dd...dd

aa...aa：エラーが発生した XA リソースのメソッド名

bb...bb：aa...aa が返したエラーコード

cc...cc：エラーが発生したトランザクションの XID（トランザクション ID）

dd...dd：保守情報

［内容］

このメッセージは，Cosminexus Component Container のメッセージログに出力されます。XA リソースでエラーが発生したため，トランザクションはヒューリスティックに決着されます。このため，トランザクションで処理したデータの一貫性が失われる可能性があります。

［システムの処置］

XA リソースのリターンコードに応じて適切な処理を行います。

［対策］

トランザクションで処理したデータの一貫性が失われていないかどうか確認してください。また，ご使用のリソースマネージャのマニュアルを参照して，原因を取り除いてください。

KFCB32401-E

The transaction was heuristically committed and rolled back due to one or more XAResource errors. method=aa...xid= bb...bb

aa...aa：エラーが発生した XA リソースのメソッド名

bb...bb：エラーが発生したトランザクションの XID

［内容］

このメッセージは，Cosminexus Component Container のメッセージログに出力されます。一つ以上の XA リソースのエラーが原因で，トランザクションは部分的にコミットおよびロールバックされ，データの一貫性が失われました。

［システムの処置］

このトランザクションに関連するすべての XA リソースのリターンコードに応じて，それぞれ適切な処理を実行します。

［対策］

ご使用のリソースマネージャのマニュアルを参照して，原因を取り除いてください。

KFCB5nnnn-X

Error reason=xx...xx code=yy...yy

KFCB50000-X ～ KFCB59999-X まではユーザが作成するメッセージです。詳細は「6.7.1　UAP ログの出力方式」を参照してください

12 Java OTS が出力するメッセージ

この章では，「11.2　メッセージ一覧」に記載されている KFCB32206-E メッセージ（Java OTS が出力するメッセージ）の形式および詳細を説明します。

12.1 メッセージの形式

この節では，メッセージの形式や出力先について説明します。

12.1.1 メッセージの出力形式

メッセージの出力形式については，「11.1.1 メッセージの出力形式」を参照してください。

12.1.2 メッセージの記述形式

この章のメッセージの記述形式を次に示します。

メッセージ ID

メッセージテキスト

説明

［内容］

例外の意味を示します。

［システムの処理］

システムがメッセージを出力したあとにする主な処理を示します。

［対策］

メッセージ確認時の TPBroker 管理者の処置を示します。

Cosminexus TPBroker の場合※

［内容］

例外の意味を示します。

［システムの処理］

システムがメッセージを出力したあとにする主な処理を示します。

［対策］

メッセージ確認時の TPBroker 管理者の処置を示します。

注※

Cosminexus TPBroker だけに発生する例外の場合，または Cosminexus TPBroker 特有の情報がある例外の場合に記述しています。

12.1.3 例外

TPBroker は，自分自身で発生させた CORBA SystemException（org.omg.CORBA.SystemException），または CORBA 標準仕様の User Exception（org.omg.CORBA.UserException）にエラーの説明を組み込みます。ORB およびサーバ側で発生した CORBA SystemException には詳細情報が提供されません。詳細情報を得

るには，次の例のようにコーディングして
org.omg.CORBA.SystemException.toString() を呼び出してください。

```
try{
     current.begin();
} catch(org.omg.CORBA.SystemException e) {
     System.out.println("Exception Catch:" + e);
}
```

形式は，各 CORBA SystemException で定義されています。

例

- org.omg.CORBA.BAD_PARAM

不正なパラメタが指定されています。

12.1.4　メッセージの出力先

メッセージの出力先は，標準出力および標準エラー出力です。標準エラー出力には，メモリ不足など，例外を発生させられない場合にメッセージが出力されます。

12.2　メッセージ一覧

この節では，メッセージを出力先別に示します。なお，この節で示す Java OTS が出力するメッセージのメッセージ ID はすべて KFCB32206-E です。

12.2.1　SystemException および UserException に組み込まれるメッセージ

ここでは，SystemException および UserException に組み込まれるメッセージを説明します。なお，次に示す例外は，Cosminexus TPBroker だけに発生する例外，または Cosminexus TPBroker 特有の情報がある例外です。

- java.lang.IllegalArgumentException 例外
- java.lang.IllegalStateException 例外
- org.omg.CORBA.COMM_FAILURE 例外
- org.omg.CORBA.IMP_LIMIT 例外
- org.omg.CORBA.INITIALIZE 例外
- org.omg.CORBA.INVALID_TRANSACTION 例外
- org.omg.CORBA.NO_IMPLEMENT 例外
- org.omg.CORBA.NO_PERMISSION 例外
- org.omg.CORBA.OBJECT_NOT_EXIST 例外
- org.omg.CORBA.PolicyError 例外
- org.omg.CORBA.TRANSACTION_MODE 例外
- org.omg.CORBA.TRANSACTION_REQUIRED 例外
- org.omg.CORBA.TRANSACTION_ROLLEDBACK 例外
- org.omg.CORBA.TRANSACTION_UNAVAILABLE 例外
- org.omg.PortableServer.POAPackage.InvalidPolicy 例外

(1)　java.lang.IllegalArgumentException 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：不正なパラメタ名

bb...bb：メソッド名

Cosminexus TPBroker の場合

[内容]
　不正なパラメタが指定されました。

[システムの処理]
　メソッドオペレーションを停止します。

［対策］
保守員に連絡してください。

(2) java.lang.IllegalStateException 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：理由コード

bb...bb：メソッド名

Cosminexus TPBroker の場合

［内容］
このオペレーションは許可されていません。

［システムの処理］
オペレーションを停止します。

［対策］
保守員に連絡してください。

(3) org.omg.CORBA.BAD_PARAM 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：不正なパラメタ名

bb...bb：メソッド名

［内容］
不正なパラメタが指定されました。

［システムの処理］
メソッドオペレーションを停止します。

［対策］
パラメタの値を調べてください。

(4) org.omg.CORBA.COMM_FAILURE 例外

KFCB32206-E

Error reason=TransactionContextServer code=aa...aa

aa...aa：メソッド名

Cosminexus TPBroker **の場合**

［内容］

トランザクションコンテキストサーバとの通信でエラーが発生しました。

［システムの処置］

オペレーションを停止します。

［対策］

トランザクションコンテキストサーバが正常に動作しているかどうかを確認してください。

(5) org.omg.CORBA.IMP_LIMIT 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：理由コード

Time Independent Invocation

Time Independent Invocation はサポートされていません。

One Way Method Invocation

トランザクションコンテキストの暗黙的プロパゲーションによる一方送信（一方受信）呼び出しはサポートされていません。

An error occurred in TCS execution environment

TPBroker 環境でエラーが発生しました。

bb...bb：メソッド名

［内容］

TPBroker でサポートしていないケースが発生しました。

［システムの処理］

オペレーションを停止します。

［対策］

サーバの呼び出し方法を見直してください。

Cosminexus TPBroker **の場合**

［内容］

実装の上限を超えました。または，サポートしていない機能を使用しています。

［システムの処理］

オペレーションを停止します。

［対策］

TPBroker 環境でエラーが発生している場合は，syslog（UNIX），またはイベントログ（Windows）を参照して原因を取り除いてください。その他の場合は，アプリケーションプログラムの処理を見直してください。

vmcid=0x48542000 minor code=257

Cosminexus TPBroker の場合

［内容］

最大接続可能 CRM ブランチ数が上限を超えています。

［システムの処置］

メソッドオペレーションを停止します。

［対策］

トランザクション定義 /OTS/TM max_crm_branch_count の指定値を調整し，TPBroker を再起動してください。

(6) org.omg.CORBA.INITIALIZE 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：理由コード

Already Init Class Is Loaded：すでに Init クラスがロードされています。

Already XARInit Class Is Loaded：すでに XARInit クラスがロードされています。

bb...bb：メソッド名

［内容］

OTS 起動プロパティの指定が誤っています。

［システムの処理］

アプリケーションプログラムの起動を中止します。

［対策］

適切な起動プロパティを設定してアプリケーションプログラムを起動してください。

Cosminexus TPBroker の場合

［内容］

OTS 起動プロパティの指定が誤っています。

［システムの処置］

アプリケーションプログラムの起動を中止します。

［対策］

Cosminexus TPBroker では，OTS 起動プロパティを指定できません。

(7) org.omg.CORBA.INTERNAL 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：理由コード

NO_MEMORY：メモリ不足

EXCEPTION：不正な例外の発生

INTERNAL：内部エラーの発生

bb...bb：保守情報

［内容］
内部エラーが発生しました。

［システムの処理］
オペレーションを停止します。

［対策］
理由コードが NO_MEMORY の場合，必要に応じてメモリ確保を増やしてください。解決できない場合，保守員に連絡してください。

(8) org.omg.CORBA.INV_OBJREF 例外

KFCB32206-E

Error reason=Invalid Server's TaggedComponent code=aa...aa

aa...aa：メソッド名

［内容］
サーバ IOR の OTS 関連 TaggedComponent が不正です。

［システムの処理］
オペレーションを停止します。

［対策］
サーバの処理を見直してください。

(9) org.omg.CORBA.INVALID_TRANSACTION 例外

KFCB32206-E

Error reason=Server's OTSPolicy: FORBIDS code=aa...aa

aa...aa：メソッド名

［内容］

サーバはトランザクションコンテキストの暗黙的プロパゲーションをしない呼び出しを要求しています。

［システムの処理］
オペレーションを停止します。

［対策］
サーバに対し，トランザクションコンテキストの暗黙的プロパゲーションをしない呼び出しをしているかどうかを調べてください。

Cosminexus TPBroker の場合

［内容］
サーバはトランザクショナルではない呼び出しを要求しています。

［システムの処理］
オペレーションを停止します。

［対策］
トランザクショナルな呼び出しをしていないかどうかを調べてください。

(10) org.omg.CORBA.MARSHAL 例外

KFCB32206-E

Error reason=Invalid ServiceContext code=aa...aa

aa...aa：メソッド名

［内容］
OTS サービスコンテキストのマーシャリングに失敗しました。

［システムの処理］
オペレーションを停止します。

［対策］
クライアントの処理を見直してください。

(11) org.omg.CORBA.NO_IMPLEMENT 例外

KFCB32206-E

Error reason=Not Supported (Interface) code=aa...aa

aa...aa：インタフェース名

Cosminexus TPBroker の場合

［内容］
サポートしていないインタフェースです。

［システムの処置］

オペレーションを停止します。

［対策］

アプリケーションプログラムの処理を見直してください。

(12) org.omg.CORBA.NO_MEMORY 例外

KFCB32206-E

Error reason=NO_MEMORY code=aa...aa

aa...aa：保守情報

［内容］

メモリ不足が発生しています。

［システムの処理］

オペレーションを停止します。

［対策］

メモリサイズを増やすか，または不要なメモリ確保をしていないかどうかを見直してください。

(13) org.omg.CORBA.NO_PERMISSION 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：理由コード

trn_ctx_sv_name

トランザクションコンテキストサーバと接続したあとに，名前を指定できません。

Branch Transaction

このオペレーションはトランザクションブランチに許可されていません。

Inactive Transaction

このトランザクションは Active ではありません。

recover permission check failed

XAResource.recover を正しく発行できないリソースを使用しようとしています。

not successfully recovered yet

システム再起動後の回復処理が完了していないリソースを使用しようとしています。

Not Registrable XAResource Object

登録できない XAResource オブジェクトです。

Not Supported TCS

現在接続しているトランザクションコンテキストサーバは，サポート外です。

bb...bb：メソッド名

［内容］

このオペレーションは許可されていません。

［システムの処理］

オペレーションを停止します。

［対策］

ユーザアプリケーションプログラムのオペレーションを調べてください。

Cosminexus TPBroker の場合

［内容］

このオペレーションは許可されていません。

［システムの処理］

オペレーションを停止します。

［対策］

理由コードに応じて次に示す対策をしてください。

Inactive Transaction

トランザクションタイムアウトが発生するとトランザクションの状態が Active でなくなり，この例外が発生する場合があります。タイムアウトの設定値を見直してください。

recover permission check failed

XAResource.recover の発行に失敗している場合は，リソースの設定を見直してください。

not successfully recovered yet

システム再起動後の回復処理が完了していないリソースを使用した場合，間隔をおいて再試行してください。

Not Registrable XAResource Object

登録したい XAResource オブジェクトが指定しているリソースマネージャを確認してください。すでに OTS に登録されている XAResource オブジェクトが指定するリソースマネージャと同じリソースマネージャを指定している場合，OTS は登録を許可しません。XARCoordinator.registerXAResource の仕様を確認して，再度登録してください。

Not Supported TCS

現在使用している Cosminexus でバンドルされているトランザクションコンテキストサーバを使用しているか確認してください。

上記以外の理由コードの例外が発生した場合は，保守員に連絡してください。

(14) org.omg.CORBA.OBJECT_NOT_EXIST 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：理由コード

Already Transaction Is Completed
　トランザクションは，すでに決着しています。

Transaction is NullTx
　NullTx トランザクションです。

bb...bb：メソッド名

Cosminexus TPBroker の場合

［内容］
無効なトランザクションコンテキストを操作しようとしています。

［システムの処置］
オペレーションを停止します。

［対策］
無効になったトランザクションコンテキストは，それ以降ユーザプログラムでは使用しないようにしてください。

(15) org.omg.CORBA.PolicyError 例外

KFCB32206-E

Error reason=aa...aa: bb...bb code=cc...cc

aa...aa：不正な値が設定されたポリシータイプ
OTSPolicy：OTSPolicy の値が不正です。
InvocationPolicy：InvocationPolicy の値が不正です。
NonTxTargetPolicy：NonTxTargetPolicy の値が不正です。

bb...bb：設定された値

cc...cc：メソッド名

［内容］
不正な値が設定されました。

［システムの処理］
オペレーションを停止します。

［対策］
設定した値を調べてください。

Cosminexus TPBroker の場合

［内容］
不正な値が設定されました。

［システムの処理］
オペレーションを停止します。

［対策］
保守員に連絡してください。

(16) org.omg.CORBA.TRANSACTION_MODE 例外

KFCB32206-E

Error reason=Server's InvocationPolicy: aa...aa code=bb...bb

aa...aa：InvocationPolicy の値

bb...bb：メソッド名

［内容］
サーバの呼び出し方法が InvocationPolicy の値に従っていません。

［システムの処理］
オペレーションを停止します。

［対策］
サーバの呼び出し方法，および InvocationPolicy の値を調べてください。

Cosminexus TPBroker の場合

［内容］
サーバの呼び出し方法が InvocationPolicy の値に従っていません。

［システムの処理］
オペレーションを停止します。

［対策］
InvocationPolicy の値を調べてください。

(17) org.omg.CORBA.TRANSACTION_REQUIRED 例外

KFCB32206-E

Error reason=Server's OTSPolicy: REQUIRES code=aa...aa

aa...aa：メソッド名

［内容］
サーバはトランザクションコンテキストの暗黙的プロパゲーションによる呼び出しを要求しています。

［システムの処理］

オペレーションを停止します。

［対策］

サーバに対し，トランザクションコンテキストの暗黙的プロパゲーションによる呼び出しをしているかどうかを調べてください。

Cosminexus TPBroker の場合

［内容］

サーバはトランザクショナルな呼び出しを要求しています。

［システムの処理］

オペレーションを停止します。

［対策］

トランザクショナルな呼び出しをしているかどうかを調べてください。

(18) org.omg.CORBA.TRANSACTION_ROLLEDBACK 例外

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：理由コード

Already Transaction Is Timedout

トランザクションはすでにタイムアウトしています。

TransactionContextServer

処理中にトランザクションコンテキストサーバと接続できませんでした。

An error occurred in TCS execution environment

TPBroker 環境でエラーが発生しました。

bb...bb：メソッド名

［内容］

トランザクションはすでにロールバックしたか，またはロールバックすることに決定しました。

［システムの処置］

メソッドオペレーションを停止します。

［対策］

TPBroker 環境でエラーが発生している場合は，syslog（UNIX），またはイベントログ（Windows）を参照して原因を取り除いてください。

Cosminexus TPBroker の場合

［内容］

トランザクションはすでにロールバックしたか，またはロールバックすることに決定しました。

［システムの処置］

メソッドオペレーションを停止します。

［対策］

すでにトランザクションのロールバックが決定したトランザクションコンテキストは，それ以降ユーザプログラムでは使用しないようにしてください。
TPBroker 環境でエラーが発生している場合は，syslog（UNIX），またはイベントログ（Windows）を参照して原因を取り除いてください。

(19) org.omg.CORBA.TRANSACTION_UNAVAILABLE 例外

KFCB32206-E

Error reason=TransactionContextServer code=aa...aa

aa...aa：メソッド名

［内容］

トランザクションコンテキストサーバとの通信でエラーが発生しました。

［システムの処理］

オペレーションを停止します。

［対策］

OSAgent ポートなどの実行環境が正しく設定されているかどうか，またはトランザクションコンテキストサーバが正常に動作しているかどうかを確認してください。
また，ご使用の TPBroker のバージョンに適合するトランザクションコンテキストサーバを使用しているかどうかを確認してください。

Cosminexus TPBroker の場合

［内容］

トランザクションコンテキストサーバとの通信でエラーが発生しました。

［システムの処理］

オペレーションを停止します。

［対策］

TPBroker に設定した OSAGENT_PORT 環境変数と，J2EE サーバの vbroker.agent.port プロパティの設定が一致しているかどうかなど，実行環境が正しく設定されているかどうか，またはトランザクションコンテキストサーバが正常に動作しているかどうかを確認してください。また，ご使用の TPBroker のバージョンに適合するトランザクションコンテキストサーバを使用しているかどうかを確認してください。

(20) org.omg.PortableServer.POAPackage.InvalidPolicy 例外

KFCB32206-E

Error reason=OTSPolicy:aa...aa, InvocationPolicy:bb...bb code=cc...cc

aa...aa：設定された OTSPolicy の値

bb...bb：設定された InvocationPolicy の値

cc...cc：メソッド名

［内容］
OTSPolicy の値と InvocationPolicy の値の組み合わせが不正です。

［システムの処理］
オペレーションを停止します。

［対策］
OTSPolicy の値と InvocationPolicy の値の組み合わせを調べてください。

Cosminexus TPBroker の場合

［内容］
OTSPolicy の値と InvocationPolicy の値の組み合わせが不正です。

［システムの処理］
オペレーションを停止します。

［対策］
保守員に連絡してください。

KFCB32206-E

Error reason=Duplicate Policy code=aa...aa

aa...aa：メソッド名

［内容］
OTSPolicy，または InvocationPolicy を重複して設定しています。

［システムの処理］
オペレーションを停止します。

［対策］
OTSPolicy，または InvocationPolicy を重複して設定していないかどうかを調べてください。

Cosminexus TPBroker の場合

［内容］
OTSPolicy，または InvocationPolicy を重複して設定しています。

［システムの処理］
オペレーションを停止します。

［対策］

保守員に連絡してください。

12.2.2　標準エラー出力に出力されるメッセージ

KFCB32206-E

Error reason=aa...aa code=bb...bb

aa...aa：理由コード

NO_MEMORY：メモリ不足

EXCEPTION：不正な例外の発生

INTERNAL：内部エラーの発生

bb...bb：保守情報

［内容］

エラーが発生しました。

［システムの処理］

オペレーションを停止します。

［対策］

理由コードが NO_MEMORY の場合，メモリサイズを増やすか，または不要なメモリ確保をしていないかどうかを見直してください。
解決できない場合，保守員に連絡してください。

索引

A

C

I

J

L

N

O

P

R

S

T

U

V

W

X

あ

い

う

お

か

す

せ

ち

つ

て

と

ね

ひ

ふ

へ

ほ

ま

め

ら

り

ろ

ソフトウェアマニュアルのサービス ご案内

1．マニュアル情報ホームページ

ソフトウェアマニュアルの情報をインターネットで公開しています。

URL　http://www.hitachi.co.jp/soft/manual/

ホームページのメニューは次のとおりです。

■マニュアル一覧	日立コンピュータ製品マニュアルを製品カテゴリ，マニュアル名称，資料番号のいずれかから検索できます。
■CD-ROMマニュアル	日立ソフトウェアマニュアルと製品群別CD-ROMマニュアルの仕様について記載しています。
■マニュアルのご購入	マニュアルご購入時のお申し込み方法を記載しています。
■オンラインマニュアル	一部製品のマニュアルをインターネットで公開しています。
■サポートサービス	ソフトウェアサポートサービスお客様向けページでのマニュアル公開サービスを記載しています。
■ご意見・お問い合わせ	マニュアルに関するご意見，ご要望をお寄せください。

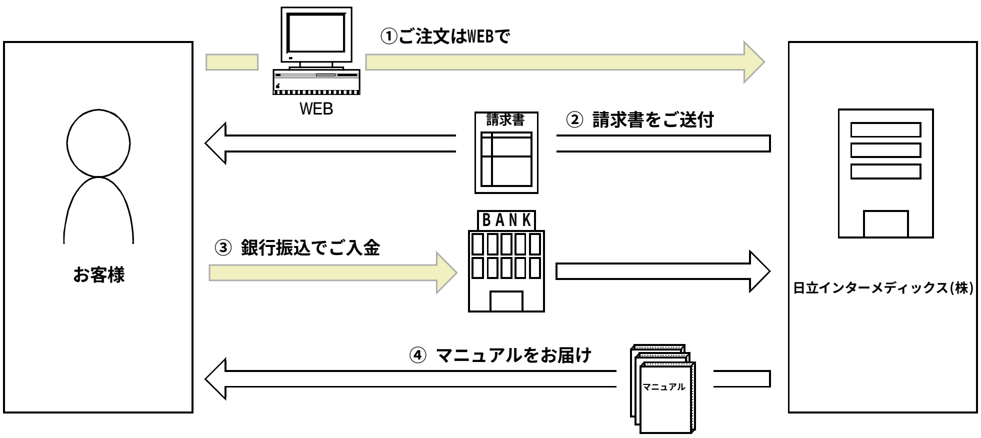
2．インターネットでのマニュアル公開

2種類のマニュアル公開サービスを実施しています。

(1) マニュアル情報ホームページ「オンラインマニュアル」での公開
　製品をよりご理解いただくためのご参考として，一部製品のマニュアルを公開しています。

(2) ソフトウェアサポートサービスお客様向けページでのマニュアル公開
　ソフトウェアサポートサービスご契約のお客様向けにマニュアルを公開しています。公開しているマニュアルの一覧，本サービスの対象となる契約の種別などはマニュアル情報ホームページの「サポートサービス」をご参照ください。

3．マニュアルのご注文



① マニュアル情報ホームページの「マニュアルのご購入」にアクセスし，お申し込み方法をご確認のうえWEBからご注文ください。ご注文先は日立インターメディックス(株)となります。

② ご注文いただいたマニュアルについて請求書をお送りします。

③ 請求書の金額を指定銀行へ振り込んでください。

④ 入金確認後7日以内にお届けします。在庫切れの場合は，納期を別途ご案内いたします。